Ektron

Redefining Web Content Management

# Ektron CMS400.NET Developer Reference Manual

**Version 5.0, Revision 8**

Ektron® Inc.
5 Northern Blvd., Suite 6
Amherst, NH 03031
Tel: +1 603-594-0249
Fax: +1 603-594-0258
Email: sales@ektron.com
http://www.ektron.com

Copyright 2005 Ektron®, Inc. All rights reserved.

EKTRON is a registered trademark of Ektron, Inc.

Version 5.0, Revision 8, July 22, 2005

For updated versions of this manual, please visit:
http://www.ektron.com/cms400-web-cms.aspx?id=1701#documentation

# EKTRON, INC. SOFTWARE LICENSE AGREEMENT

## ** READ THIS BEFORE LOADING SOFTWARE**

YOUR RIGHT TO USE THE PRODUCT DELIVERED IS SUBJECT TO THE TERMS AND CONDITIONS SET OUT IN THIS LICENSE AGREEMENT. USING THIS PRODUCT SIGNIFIES YOUR AGREEMENT TO THESE TERMS. IF YOU DO NOT AGREE TO THIS SOFTWARE LICENSE AGREEMENT, DO NOT DOWNLOAD.

CUSTOMER should carefully read the following terms and conditions before using the software program(s) contained herein (the "Software"). Downloading and/or using the Software or copying the Software onto CUSTOMER'S computer hard drive indicates CUSTOMER'S acceptance of these terms and conditions. If CUSTOMER does not agree with the terms of this agreement, CUSTOMER should not download.

Ektron, Inc. ("Ektron") grants, and the CUSTOMER accepts, a nontransferable and nonexclusive License to use the Software on the following terms and conditions:

1. Right to use: The Software is licensed for use only in delivered code form. Each copy of the Software is licensed for use only on a single URL. Each license is valid for the number of seats listed below (the "Basic Package"). Any use of the Software beyond the number of authorized seats contained in the Basic Package without paying additional license fees as provided herein shall cause this license to terminate. Should CUSTOMER wish to add seats beyond the seats licensed in the Basic Package, the CUSTOMER may add seats on a block basis at the then current price for additional seats (see product pages for current price). The Basic Packages are as follows:

Ektron eMPower for ColdFusion -- Licensed for ten seats (10 named users) per URL.

Ektron CMS100 -- Licensed for five seats (5 named users) per URL.

Ektron CMS200 -- Licensed for ten seats (10 named users) per URL.

Ektron CMS300 -- Licensed for ten seats (10 named users) per URL.

Ektron CMS400 -- Licensed for ten seats (10 named users) per URL.

Ektron eWebEditPro -- Licensed for ten seats (10 named users) per URL.

Ektron eWebEditPro+XML — Licensed for ten seats (10 named users) per URL.

Ektron eWebWP — Licensed for unlimited seats per URL.

For purposes of this section, the term "seat" shall mean an individual user provided access to the capabilities of the Software.

The CUSTOMER may not modify, alter, reverse engineer, disassemble, or decompiler the Software. This software product is licensed, not sold.

2. Duration: This License shall continue so long as CUSTOMER uses the Software in compliance with this License. Should CUSTOMER breach any of its obligations hereunder, CUSTOMER agrees to return all copies of the Software and this License upon notification and demand by Ektron.

3. Copyright: The Software (including any images, "applets," photographs, animations, video, audio, music and text incorporated into the Software) as well as any accompanying written materials (the "Documentation") is owned by Ektron or its suppliers, is protected by United States copyright laws and international treaties, and contains confidential information and trade secrets. CUSTOMER agrees to protect the confidentiality of the Software and Documentation. CUSTOMER agrees that it will not provide a copy of this Software or Documentation nor divulge any proprietary information of Ektron to any person, other than its employees, without the prior consent of Ektron; CUSTOMER shall use its best efforts to see that any user of the Software licensed hereunder complies with this license.

4. Limited Warranty: Ektron warrants solely that the medium upon which the Software is delivered will be free from defects in material and workmanship under normal, proper and intended usage for a period of three (3) months from the date of receipt. Ektron does not warrant the use of the Software will be uninterrupted or error free, nor that program errors will be corrected. This limited warranty shall not apply to any error or failure resulting from (i) machine error, (ii) Customer's failure to follow operating instructions, (iii) negligence or accident, or (iv) modifications to the Software by any person or entity other than Company. In the event of a breach of warranty, Customer's sole and exclusive remedy, is repair of all or any portion of the Software. If such remedy fails of its essential purpose, Customer's sole remedy and Ektron's maximum liability shall be a refund of the paid purchase price for the defective Products only. This limited warranty is only valid if Ektron receives written notice of breach of warranty within thirty days after the warranty period expires.

5. Limitation of Warranties and Liability: THE SOFTWARE AND DOCUMENTATION ARE SOLD "AS IS" AND WITHOUT ANY WARRANTIES AS TO THE PERFORMANCE, MERCHANTIBILITY, DESIGN, OR OPERATION OF THE SOFTWARE. NO WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE IS OFFERED. EXCEPT AS DESCRIBED IN SECTION 4, ALL WARRANTIES EXPRESS AND IMPLIED ARE HEREBY DISCLAIMED.

NEITHER COMPANY NOR ITS SUPPLIERS SHALL BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS OR GOODWILL, LOSS OF DATA OR USE OF DATA, INTERRUPTION OF BUSINESS NOR FOR ANY OTHER INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY KIND UNDER OR ARISING OUT OF, OR IN ANY RELATED TO THIS AGREEMENT, HOWEVER,

CAUSED, WHETHER FOR BREACH OF WARRANTY, BREACH OR REPUDIATION OF CONTRACT, TORT, NEGLIGENCE, OR OTHERWISE, EVEN IF COMPANY OR ITS REPRESENTATIVES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH LOSS.

6. Miscellaneous: This License Agreement, the License granted hereunder, and the Software may not be assigned or in any way transferred without the prior written consent of Ektron. This Agreement and its performance and all claims arising from the relationship between the parties contemplated herein shall be governed by, construed and enforced in accordance with the laws of the State of New Hampshire without regard to conflict of laws principles thereof. The parties agree that any action brought in connection with this Agreement shall be maintained only in a court of competent subject matter jurisdiction located in the State of New Hampshire or in any court to which appeal therefrom may be taken. The parties hereby consent to the exclusive personal jurisdiction of such courts in the State of New Hampshire for all such purposes. The United Nations Convention on Contracts for the International Sale of Goods is specifically excluded from governing this License. If any provision of this License is to be held unenforceable, such holding will not affect the validity of the other provisions hereof. Failure of a party to enforce any provision of this Agreement shall not constitute or be construed as a waiver of such provision or of the right to enforce such provision. If you fail to comply with any term of this License, YOUR LICENSE IS AUTOMATICALLY TERMINATED. This License represents the entire understanding between the parties with respect to its subject matter.

YOU ACKNOWLEDGE THAT YOU HAVE READ THIS AGREEMENT, THAT YOU UNDERSTAND THIS AGREEMENT, AND UNDERSTAND THAT BY CONTINUING THE INSTALLATION OF THE SOFTWARE, BY LOADING OR RUNNING THE SOFTWARE, OR BY PLACING OR COPYING THE SOFTWARE ONTO YOUR COMPUTER HARD DRIVE, YOU AGREE TO BE BOUND BY THIS AGREEMENT'S TERMS AND CONDITIONS. YOU FURTHER AGREE THAT, EXCEPT FOR WRITTEN SEPARATE AGREEMENTS BETWEEN EKTRON AND YOU, THIS AGREEMENT IS A COMPLETE AND EXCLUSIVE STATEMENT OF THE RIGHTS AND LIABILITIES OF THE PARTIES.

Copyright 1999 - 2005 Ektron, Inc. All rights reserved.

LA10031, Revision 1.4

# What's New in the 5.0 Release

- Dreamweaver functions now look and act like standard .NET server controls. *See Also:* "Dreamweaver Extension Support" on page 203
- New Documentation: "Creating a Web Site" on page 2. This section describes creating a Web site in Microsoft Visual Studio .NET 2003 and Macromedia Dreamweaver MX 2004.
- The ability to create web sites manually using Ektron's custom ASP functions. *See Also:* "Custom ASP Functions" on page 273
- The ability to create web sites manually using Ektron's custom ColdFusion functions. *See Also:* "Custom ColdFusion Functions" on page 377
- The ability to create web sites manually using Ektron's custom PHP functions. *See Also:* "Custom PHP Functions" on page 404
- The RssAggregator Server Control lets you consume and display an RSS feed on your Web site. *See Also:* "RssAggregator Server Control" on page 194
- The BreadCrumb Server Control creates a trail of Web pages for your site visitors to follow. *See Also:* "BreadCrumb Server Control" on page 198

# Summary Table of Contents


What's New in the 5.0 Release ..... i

About This Manual ..... 1

Creating a Web Site ..... 2

- Creating a Web Application Using Microsoft Visual Studio .NET 2
- Creating a Site Using Macromedia Dreamweaver MX 2004 ..... 7

Introduction to Ektron CMS400.NET Server Controls ... 24

- Opening the Sample Project ..... 25
- Setting Up a Template ..... 26
- Making the Server Controls Available ..... 27
- Inserting Server Controls Using Drag and Drop ..... 29
- Inserting Server Controls Programmatically ..... 33
- Using Drag & Drop and Programmatically Together ..... 51
- Data Binding with Server Controls ..... 53
- Caching with Server Controls ..... 59
- List of Server Controls ..... 61

Login Server Control ..... 63

- Placing a Login Button ..... 63
- Login Server Control Properties ..... 64

LanguageSelect Server Control ..... 65

- LanguageSelect Server Control vs. LanguageAPI Server Control ..... 65
- LanguageSelect Server Control Properties ..... 66

LanguageAPI Server Control ..... 68

- LanguageSelect Server Control vs. LanguageAPI Server Control ..... 68
- LanguageAPI Server Control Properties ..... 68
- Code Behind Only Properties ..... 69

ContentBlock Server Control ..... 71

- Static Content Block ..... 71
- Dynamic Content Block ..... 73
- XML Content Block ..... 76
- Using the ContentBlock Server Control Programmatically Example ..... 79

FormBlock Server Control ..... 81
FormBlock Server Control Properties ..... 82
Automatic versus Manual Generation of Form Tags ..... 84
Search Server Control ..... 85
Search Server Control vs. IndexSearch Server Control ..... 86
Search Server Control Properties ..... 86
Specify Search Control To Display Search Results on a Separate Page ..... 94
IndexSearch Server Control ..... 97
IndexSearch Server Control vs. Search Server Control ..... 98
IndexSearch Server Control Properties ..... 98
Using the IndexSearch Server Control Programmatically Example ..... 106
Programmatically Predefined General Search Results Replaced By Specific Search Results. ..... 109
Turn Off Caching In Netscape Navigator and Mozilla Firefox. ... 113
ListSummary Server Control ..... 114
ListSummary Server Control Properties ..... 115
List Summary Example ..... 119
Collection Server Control ..... 120
Collection Server Control Properties ..... 121
Using the Collection Server Control Programmatically Example ..... 124
Calendar Server Control ..... 127
Calendar Server Control Properties ..... 129
Calendar Style Sheet ..... 133
Menu Server Control ..... 161
Menu Server Control vs. DhtmlMenu Server Control ..... 161
Menu Server Control Properties ..... 162
Using DisplayXslt Samples ..... 164
DhtmlMenu Server Control ..... 178
Menu Server Control vs. DhtmlMenu Server Control ..... 179
DhtmlMenu Server Control Properties ..... 179
MetaData Server Control ..... 186
MetaData Server Control vs. MetaDataList Server Control .. 186
MetaData Server Control Properties ..... 186
Using the MetaData Server Control ..... 187
MetaDataList Server Control ..... 189
MetaData Server Control vs. MetaDataList Server Control .. 190
MetaDataList Server Control Properties ..... 190
RssAggregator Server Control ..... 194
RSSAggregator Server Control Properties ..... 195

Using the RssAggregator Server Control....196
BreadCrumb Server Control....198
BreadCrumb Server Control Properties....199
Using the BreadCrumb Server Control....201

Dreamweaver Extension Support....203
Requirements....203
Preparing Dreamweaver for Ektron CMS400.NET Functions....203
Installing the Extension....204
Setup....208
Using the Ektron CMS400.NET Extension....210
Inserting Custom Functions....216

Standard ASP.NET Methods....263
Replacement of Standard Methods....263
Referencing Methods in Visual Basic....265
Adding a New Web Form....265
Requirements for Standard Methods....266
Standard Methods....267
Add Content....268

Custom ASP Functions....273
Login Function....276
Login Tag Example....277
Placing Your Login Button....277
Content Block Function....278
Content Block Examples....281
Search Functions....288
Search Tag Example....291
Search Tag Attributes....292
Search Display Function....299
Search Display Example....300
Search Display Attributes....301
List Summary Function....304
List Summary Example....306
List Summary Attributes....307
XML List Summary Function....312
Archive List Summary Function....317
Single Summary Function....318
Single Summary Example....319
Single Summary Attributes....319

RSS Summary Function ..... 322
RSS Summary Example ..... 323
RSS Summary Attributes ..... 326
Collection Function ..... 328
Collection Examples ..... 328
Collection Attributes ..... 333
Creating Custom Display Functions ..... 334
RSS Collection Function ..... 341
XML Collection Function ..... 343
Random Content ..... 348
Random Summary ..... 349
Event Calendar Function ..... 350
Event Calendar Style Sheet ..... 350
Event Calendar Sample ..... 351
Calendar Function ..... 353
Calendar Style Sheet ..... 355
Event Calendar Sample ..... 355
Form Function ..... 357
MultiPurpose Function ..... 360
Metadata Function ..... 361
Creating an Item List ..... 361
ecmMetadata Example ..... 362
More ecmMetadata Examples ..... 364
ecmMetadata Attributes ..... 366
Show Site Language Function ..... 369
DHTML Menus ..... 370
Additional Documentation in Ektron Knowledge Base ..... 370
Displaying a Menu on a Web Page ..... 370
Customizing the Menu's Appearance ..... 371
ek_Menu.js ..... 372
XML Menu ..... 375

Custom ColdFusion Functions ..... 377
Login Function ..... 379
Content Block Function ..... 380
Random Content Function ..... 381
Random Summary Function ..... 382
Search Function ..... 382
Search Display Function ..... 383
List Summary Function ..... 384
XML List Summary Function ..... 385
RSS List Summary Function ..... 386
Archive List Summary Function ..... 387

Single Summary Function ..... 387
Collection Function ..... 388
XML Collection Function ..... 393
RSS Collection Function ..... 394
Event Calendar Function ..... 394
Form Function ..... 397
MultiPurpose Function ..... 398
Metadata Function ..... 398
DHTML Menus ..... 401
XML Menu Function ..... 402
Language Select Function ..... 403

Custom PHP Functions ..... 404
Login Function ..... 407
Content Block Function ..... 407
Search Function ..... 408
Search Display Function ..... 409
List Summary Function ..... 410
Archive List Summary Function ..... 411
Single Summary Function ..... 411
XML List Summary Function ..... 412
RSS List Summary Function ..... 413
Collection Function ..... 413
XML Collection Function ..... 417
RSS Collection Function ..... 418
Random Content Function ..... 418
Random Summary Function ..... 419
Event Calendar Function ..... 419
Form Function ..... 422
MultiPurpose Function ..... 423
Metadata Function ..... 423
DHTML Menus ..... 426
XML Menu Function ..... 428
Language Select Function ..... 428
Custom API ..... 430

Web Services ..... 431
Introduction to Web Services ..... 431
WS Components and CMS400.NET Architecture ..... 432
Software and Hardware Requirements ..... 432
Installation of Files ..... 433
Setup Instructions ..... 433
Providing Web Service Information in Your CMS ..... 435

Creating a Page that Provides a Web Service ..... 452

Syndicating Web Content as RSS ..... 454
What is RSS Syndication? ..... 454
RSS Syndication in Ektron CMS400.NET ..... 455
Syndicating via .NET ..... 456
Architecture of RSS Syndication ..... 456
Creating Your Own aspx Collection Page ..... 456
Creating Your Own aspx List Summary Page ..... 457

Learning About Visual Studio .NET ..... 459
Grid Layout vs. Flow Layout ..... 459

Additional Information ..... 461
Supporting Documents ..... 461
Support ..... 461
Sales ..... 461
Information ..... 461

# Detailed Table of Contents


What's New in the 5.0 Release ..... i

About This Manual ..... 1

Creating a Web Site ..... 2

Creating a Web Application Using Microsoft Visual Studio .NET 2

Creating a Web Project ..... 4

Steps to Create a Web Project ..... 4

Using Web Forms ..... 6

Creating a Site Using Macromedia Dreamweaver MX 2004 ..... 7

Defining a Local Site ..... 9

Steps to Define a Folder for Use as a Local Site ..... 9

Building the Site Map in the Files Window ..... 12

Steps to Add a Home Page ..... 14

Creating a Subfolder and Files ..... 17

Steps to Create a Subfolder and Files ..... 17

Creating the Template ..... 17

Steps to Create a Template ..... 18

Applying the Template ..... 23

Steps to Apply the Template ..... 23

Testing and Viewing the Site ..... 23

Steps to Test and View the Site ..... 23

Introduction to Ektron CMS400.NET Server Controls ... 24

Opening the Sample Project ..... 25

Setting Up a Template ..... 26

Making the Server Controls Available ..... 27

Removing the Server Controls ..... 29

Updating the Server Controls ..... 29

Inserting Server Controls Using Drag and Drop ..... 29

Adding a Server Control to a Template ..... 29

ServerControlWS.asmx ..... 31

Using CMS Explorer to Browse Your Ektron CMS400.NET Site ... 32

Build the Project ..... 33

Inserting Server Controls Programmatically ..... 33

Steps to Insert a Server Control Programmatically ..... 34

Step 1: Declare the Namespace ..... 35

Step 2: Create an Instance of the New Control ..... 35
Step 3: Set the Properties ..... 36
Step 4: Call the Fill Method ..... 37
Step 5: Use .text Property to Display the Control ..... 37
Server Control Properties in Code Behind ..... 38
Displaying Properties for a Control ..... 38
Mutual Server Control Properties in Code Behind ..... 39
Accessing Additional Properties ..... 41
Accessing Items in an Array ..... 47
Referencing the Page Property ..... 47
Customizing Server Controls ..... 48
Recognizing the Server Control Within the HTML ..... 49
Recognizing the Server Control Within the Code Behind ..... 49
Customizing the Server Control in the Code Behind ..... 50
Troubleshooting Error Creating Control Message ..... 50
**Using Drag & Drop and Programmatically Together ..... 51**
**Data Binding with Server Controls ..... 53**
Steps to Data Binding Using Drag and Drop Server Controls ..... 54
Customizing the DataGrid Table View ..... 57
**Caching with Server Controls ..... 59**
**List of Server Controls ..... 61**
**Login Server Control ..... 63**
**Placing a Login Button ..... 63**
**Login Server Control Properties ..... 64**
**LanguageSelect Server Control ..... 65**
**LanguageSelect Server Control vs. LanguageAPI Server Control ..... 65**
**LanguageSelect Server Control Properties ..... 66**
**LanguageAPI Server Control ..... 68**
**LanguageSelect Server Control vs. LanguageAPI Server Control ..... 68**
**LanguageAPI Server Control Properties ..... 68**
**Code Behind Only Properties ..... 69**
**ContentBlock Server Control ..... 71**
**Static Content Block ..... 71**
ContentBlock Server Control Properties ..... 72
**Dynamic Content Block ..... 73**
**XML Content Block ..... 76**
The OverrideXslt and DisplayXslt Properties ..... 77
**Using the ContentBlock Server Control Programmatically Example ..... 79**
**FormBlock Server Control ..... 81**
**FormBlock Server Control Properties ..... 82**
**Automatic versus Manual Generation of Form Tags ..... 84**
**Search Server Control ..... 85**

**Search Server Control vs. IndexSearch Server Control..... 86**
**Search Server Control Properties ..... 86**
Search Server Control Property Example ..... 89
Text Box Examples ..... 90
Button Image Source Examples ..... 91
Search Button Text Examples ..... 92
Display Examples ..... 93
ShowDate Examples ..... 93
**Specify Search Control To Display Search Results on a Separate Page..... 94**
**IndexSearch Server Control..... 97**
**IndexSearch Server Control vs. Search Server Control..... 98**
**IndexSearch Server Control Properties..... 98**
SearchParmXML Property..... 103
**Using the IndexSearch Server Control Programmatically Example ..... 106**
**Programmatically Predefined General Search Results Replaced By Specific Search Results. ..... 109**
**Turn Off Caching In Netscape Navigator and Mozilla Firefox.... 113**
**ListSummary Server Control ..... 114**
**ListSummary Server Control Properties ..... 115**
OrderKey Property.....118
**List Summary Example ..... 119**
**Collection Server Control..... 120**
**Collection Server Control Properties..... 121**
Example of ecmNavigation Display ..... 122
ecmNavigation XSL code..... 123
Example of ecmTeaser Display ..... 123
ecmTeaser XSL code..... 124
**Using the Collection Server Control Programmatically Example..... 124**
**Calendar Server Control..... 127**
**Calendar Server Control Properties..... 129**
**Calendar Style Sheet ..... 133**
calendarStyles.css Class Description..... 133
Single Day View (Day) ..... 133
Month View (month) ..... 137
Inline View (inline) ..... 149
Month-Up with Event (monthupwithevent) ..... 151
Event Type Control (used in month display) ..... 158
**Menu Server Control..... 161**
**Menu Server Control vs. DhtmlMenu Server Control..... 161**
**Menu Server Control Properties..... 162**
**Using DisplayXslt Samples..... 164**

SampleMenu ..... 164
SampleMenu XSL code ..... 165
TreeMenu ..... 167
TreeMenu XSL code ..... 169
**DhtmlMenu Server Control ..... 178**
**Menu Server Control vs. DhtmlMenu Server Control ..... 179**
**DhtmlMenu Server Control Properties ..... 179**
Using a Style Sheet ..... 183
**MetaData Server Control ..... 186**
**MetaData Server Control vs. MetaDataList Server Control ..... 186**
**MetaData Server Control Properties ..... 186**
**Using the MetaData Server Control ..... 187**
**MetaDataList Server Control ..... 189**
**MetaData Server Control vs. MetaDataList Server Control ..... 190**
**MetaDataList Server Control Properties ..... 190**
**RssAggregator Server Control ..... 194**
**RSSAggregator Server Control Properties ..... 195**
**Using the RssAggregator Server Control ..... 196**
**BreadCrumb Server Control ..... 198**
**BreadCrumb Server Control Properties ..... 199**
**Using the BreadCrumb Server Control ..... 201**

**Dreamweaver Extension Support ..... 203**
**Requirements ..... 203**
**Preparing Dreamweaver for Ektron CMS400.NET Functions ..... 203**
**Installing the Extension ..... 204**
**Setup ..... 208**
**Using the Ektron CMS400.NET Extension ..... 210**
Custom Function Symbols ..... 213
Ektron CMS400.NET Commands Menu ..... 214
**Inserting Custom Functions ..... 216**
CMS Function ..... 216
Accessing CMS Content Functions ..... 217
Login ..... 217
Inserting a Content Block ..... 220
Inserting a Collection ..... 222
Inserting a List Summary ..... 225
Inserting a Metadata Function ..... 229
Inserting a Search Function ..... 232
Inserting an Event Calendar Function ..... 235
Inserting a Form ..... 238
Inserting an XML Index Search ..... 241
Inserting Display By Meta Value ..... 246
Inserting a Standard Menu ..... 250

Inserting a DHTML Menu ..... 253
Inserting a Language Select Box ..... 258
Inserting a Language API ..... 260

**Standard ASP.NET Methods ..... 263**
**Replacement of Standard Methods ..... 263**
**Referencing Methods in Visual Basic ..... 265**
**Adding a New Web Form ..... 265**
**Requirements for Standard Methods ..... 266**
**Standard Methods ..... 267**
**Add Content ..... 268**
Add Content Example ..... 270

**Custom ASP Functions ..... 273**
**Login Function ..... 276**
**Login Tag Example ..... 277**
**Placing Your Login Button ..... 277**
**Content Block Function ..... 278**
**Content Block Examples ..... 281**
Static Content Block ..... 282
Dynamic Content Block ..... 283
XML Content Block ..... 284
The Function's Arguments ..... 285
Dynamic and Static Combination ..... 287
**Search Functions ..... 288**
**Search Tag Example ..... 291**
**Search Tag Attributes ..... 292**
Starting Folder ..... 292
Recursive ..... 292
Target Page ..... 293
Text Box ..... 293
Maximum Characters ..... 294
Button Image Source ..... 294
Button Text ..... 295
Font Face ..... 296
Font Size ..... 297
Horizontal ..... 298
Spare 1 ..... 298
**Search Display Function ..... 299**
**Search Display Example ..... 300**
Search Display with Default Content ..... 300
**Search Display Attributes ..... 301**
Max Number of Content Blocks Returned ..... 302
Style Info ..... 303
Show Date ..... 303

**List Summary Function ..... 304**
**List Summary Example ..... 306**
**List Summary Attributes ..... 307**
Starting Folder ..... 307
Recursive ..... 308
Show Summary ..... 308
Style Info ..... 309
Order By ..... 309
Ascending/Descending ..... 310
Show Info ..... 311
Max Number ..... 311
**XML List Summary Function ..... 312**
**Archive List Summary Function ..... 317**
**Single Summary Function ..... 318**
**Single Summary Example ..... 319**
**Single Summary Attributes ..... 319**
Content ID ..... 319
Show Summary ..... 320
Style Info ..... 320
Show Info ..... 321
Spare 1 ..... 321
**RSS Summary Function ..... 322**
**RSS Summary Example ..... 323**
**RSS Summary Attributes ..... 326**
Starting Folder ..... 326
Recursive ..... 326
Order By ..... 326
Ascending/Descending ..... 327
Max Number ..... 327
**Collection Function ..... 328**
**Collection Examples ..... 328**
The ecmNavigation Display Function ..... 329
The ecmTeaser Display Function ..... 330
**Collection Attributes ..... 333**
ID ..... 333
DisplayFunction ..... 333
**Creating Custom Display Functions ..... 334**
Function Structure ..... 334
Creating a Basic Display Function ..... 334
Building Blocks ..... 337
Defining Display Functions ..... 339
Using a Custom Display Function ..... 340
**RSS Collection Function ..... 341**
**XML Collection Function ..... 343**
**Random Content ..... 348**

**Random Summary ..... 349**
**Event Calendar Function ..... 350**
**Event Calendar Style Sheet ..... 350**
**Event Calendar Sample ..... 351**
**Calendar Function ..... 353**
**Calendar Style Sheet ..... 355**
**Event Calendar Sample ..... 355**
**Form Function ..... 357**
**MultiPurpose Function ..... 360**
**Metadata Function ..... 361**
**Creating an Item List ..... 361**
**ecmMetadata Example ..... 362**
**More ecmMetadata Examples ..... 364**
Using a Dynamic Metadata Tag ..... 365
**ecmMetadata Attributes ..... 366**
Item List ..... 366
Spare 1 ..... 368
**Show Site Language Function ..... 369**
**DHTML Menus ..... 370**
**Additional Documentation in Ektron Knowledge Base ..... 370**
**Displaying a Menu on a Web Page ..... 370**
GenerateMenu and DropMenu ..... 370
Displaying Menus in a Multi-language System ..... 371
**Customizing the Menu's Appearance ..... 371**
**ek_Menu.js ..... 372**
FontFamily ..... 373
menuLiteBgColor ..... 374
**XML Menu ..... 375**

**Custom ColdFusion Functions ..... 377**
**Login Function ..... 379**
Login Tag Example ..... 380
**Content Block Function ..... 380**
Static Content Block ..... 381
Dynamic Content Block ..... 381
XML Content Block ..... 381
**Random Content Function ..... 381**
**Random Summary Function ..... 382**
**Search Function ..... 382**
Search Tag Example ..... 383
**Search Display Function ..... 383**
Search Display Example ..... 384
**List Summary Function ..... 384**
List Summary Example ..... 385

**XML List Summary Function ..... 385**
XML List Summary Example ..... 386
**RSS List Summary Function ..... 386**
List Summary Example ..... 386
**Archive List Summary Function ..... 387**
**Single Summary Function ..... 387**
Single Summary Example ..... 388
**Collection Function ..... 388**
Example Using the ecmNavigation Display Function ..... 389
Example Using the ecmTeaser Display Function ..... 390
Collection Variables ..... 393
DisplayFunction ..... 393
Building Blocks ..... 393
**XML Collection Function ..... 393**
**RSS Collection Function ..... 394**
**Event Calendar Function ..... 394**
Event Calendar Sample ..... 397
**Form Function ..... 397**
**MultiPurpose Function ..... 398**
**Metadata Function ..... 398**
**DHTML Menus ..... 401**
Additional Documentation in Ektron Knowledge Base ..... 401
Displaying a Menu on a Web Page ..... 401
GenerateMenu and DropMenu ..... 401
Customizing the Menu's Appearance ..... 402
ek_Menu.js ..... 402
**XML Menu Function ..... 402**
**Language Select Function ..... 403**

## Custom PHP Functions ..... 404

**Login Function ..... 407**
Login Function Example ..... 407
**Content Block Function ..... 407**
Static Content Block ..... 408
Dynamic Content Block ..... 408
XML Content Block ..... 408
**Search Function ..... 408**
Search Function Example ..... 409
**Search Display Function ..... 409**
Search Display Function Example ..... 410
**List Summary Function ..... 410**
List Summary Example ..... 410
**Archive List Summary Function ..... 411**
**Single Summary Function ..... 411**
Single Summary Example ..... 412

**XML List Summary Function ..... 412**
**RSS List Summary Function ..... 413**
**Collection Function ..... 413**
Collection Examples ..... 414
Example Using the ecmNavigation Display Function ..... 414
Example Using the ecmTeaser Display Function ..... 415
Collection Variables ..... 417
DisplayFunction ..... 417
Building Blocks ..... 417
**XML Collection Function ..... 417**
**RSS Collection Function ..... 418**
**Random Content Function ..... 418**
**Random Summary Function ..... 419**
**Event Calendar Function ..... 419**
Event Calendar Sample ..... 422
**Form Function ..... 422**
**MultiPurpose Function ..... 423**
**Metadata Function ..... 423**
**DHTML Menus ..... 426**
Additional Documentation in Ektron Knowledge Base ..... 426
Displaying a Menu on a Web Page ..... 426
GenerateMenu and DropMenu ..... 426
Customizing the Menu's Appearance ..... 427
ek_Menu.js ..... 427
**XML Menu Function ..... 428**
**Language Select Function ..... 428**
**Custom API ..... 430**

**Web Services ..... 431**
**Introduction to Web Services ..... 431**
CMS400.NET's Implementation of Web Services ..... 432
Benefits of Using Web Services ..... 432
**WS Components and CMS400.NET Architecture ..... 432**
**Software and Hardware Requirements ..... 432**
**Installation of Files ..... 433**
**Setup Instructions ..... 433**
**Providing Web Service Information in Your CMS ..... 435**
CMS400.NET Methods ..... 435
The CMS400.NET Content Sample Page ..... 439
The Search Display Method Parameters ..... 441
Sample Output from One Method ..... 443
GetContentBlock ..... 443
The News.aspx Sample (non-XML) ..... 444
Identifying the Code Behind File ..... 444
Setting Up an Instance of Web Services ..... 445

Getting Content Blocks ..... 445
The News.aspx Sample (XML) ..... 447
Setting Up Instances of Web Services ..... 447
Getting Content Blocks ..... 447
**Creating a Page that Provides a Web Service ..... 452**

**Syndicating Web Content as RSS ..... 454**
**What is RSS Syndication? ..... 454**
The RSS Specification ..... 454
**RSS Syndication in Ektron CMS400.NET ..... 455**
**Syndicating via .NET ..... 456**
**Architecture of RSS Syndication ..... 456**
aspx pages ..... 456
Retrieving a Collection or List Summary as an RSS Feed ..... 456
**Creating Your Own aspx Collection Page ..... 456**
Example ..... 457
Remarks ..... 457
Example ..... 457
**Creating Your Own aspx List Summary Page ..... 457**
Example ..... 458
Remarks ..... 458
Example ..... 458

**Learning About Visual Studio .NET ..... 459**
**Grid Layout vs. Flow Layout ..... 459**

**Additional Information ..... 461**
**Supporting Documents ..... 461**
**Support ..... 461**
**Sales ..... 461**
**Information ..... 461**

# About This Manual

This manual can be used as a reference for the developer, or the person is who is setting up your Ektron CMS400.NET Web site.

To utilize the full potential of an Ektron CMS400.NET driven Web site, learning and using the server controls supplied by Ektron is essential. These server controls help make your site more robust and easier to maintain. For more information, see ”Introduction to Ektron CMS400.NET Server Controls” on page 24.

This document explains how to manually use custom tags for each platform supported by Ektron CMS400.NET:

- ”Standard ASP.NET Methods” on page 263
- ”Custom ASP Functions” on page 273
- ”Custom ColdFusion Functions” on page 377
- ”Custom PHP Functions” on page 404

Besides explaining how to insert the functions to your Web templates manually, this manual also explains how to create Ektron CMS400.NET templates in a WYSIWYG environment by using the Ektron supplied server controls for Visual Studio .NET or by using the Dreamweaver extension supplied with the installation. See ”Dreamweaver Extension Support” on page 192 for additional information.

Ektron CMS400.NET has also implemented support for Web Services as well as syndication using RSS. Information about configuration and use of these features is also found in the following sections:

- ”Web Services” on page 431
- ”Syndicating Web Content as RSS” on page 454

# Creating a Web Site

When creating a Web site Ektron recommends using one of the following programs.

- Microsoft Visual Studio .NET
- Macromedia Dreamweaver MX 2004

This chapter provides an overview of creating a site using each of these programs. Each product has its own niche. Microsoft Visual Studio .NET with its intellisense, separate pages for coding, Server Controls, and ease of database manipulation is geared more toward the programmer/developer.

On the other hand, Macromedia Dreamweaver MX 2004 with its easy to use customizable interface, state of the art design tools, and enhanced coding tools, is a streamlined design tool for the designer/developer.

For more information on creating a Web site using Microsoft Visual Studio .NET, see "Creating a Web Application Using Microsoft Visual Studio .NET" on page 2

For more information on creating a Web site using Macromedia Dreamweaver MX 2004, see "Creating a Site Using Macromedia Dreamweaver MX 2004" on page 7.

## Creating a Web Application Using Microsoft Visual Studio .NET

**NOTE** Some information in this section has been copied from various Microsoft Web sites. These Web sites include:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vbcon/html/vbwlkwalkthroughcreatingbasicwebformspage.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dndotnet/html/introwebforms.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vbcon/html/vbtskCreatingWebProject.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vbcon/html/vbconintroductiontowebprojects.asp

When using Microsoft Visual Studio .NET, you are creating a Web application instead of a Web site. Web Applications typically involve more programming code and can include Web services. Fundamentally Web applications and Web sites can be the same thing. Microsoft's view is Visual Studio .NET is for building applications, Web or otherwise. If you are building a simple web site, their recommendation is to use Microsoft FrontPage.

**NOTE** Ektron recommends using Microsoft Visual Studio .NET or Macromedia Dreamweaver MX 2004. Ektron has added extended functionality for both products in **Ektron CMS400.NET**.

Most Web applications include the following:

- a Web project
- references
- assembly information
- a Global.asax file
- a style sheet
- a web.config file
- one or more Web Forms

For a complete description of the above items, visit the following Microsoft Web site:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vbcon/html/vbconintroductiontowebprojects.asp

In this section, the following is explained:


- "Creating a Web Project" on page 4
- "Using Web Forms" on page 6


This section does not explain the use of **Ektron CMS400.NET** Server Controls when building a Microsoft Visual Studio .NET Web

Application. For information on **Ektron CMS400.NET** Server Controls, see "Introduction to Ektron CMS400.NET Server Controls" on page 24.

**IMPORTANT** **Ektron CMS400.NET** Server Controls are an integral part of using **Ektron CMS400.NET** and Microsoft Visual Studio .NET together. After completing this section read, "Introduction to Ektron CMS400.NET Server Controls" on page 24.

## Creating a Web Project

The Web server on which your project exists can be on either a local or remote computer.

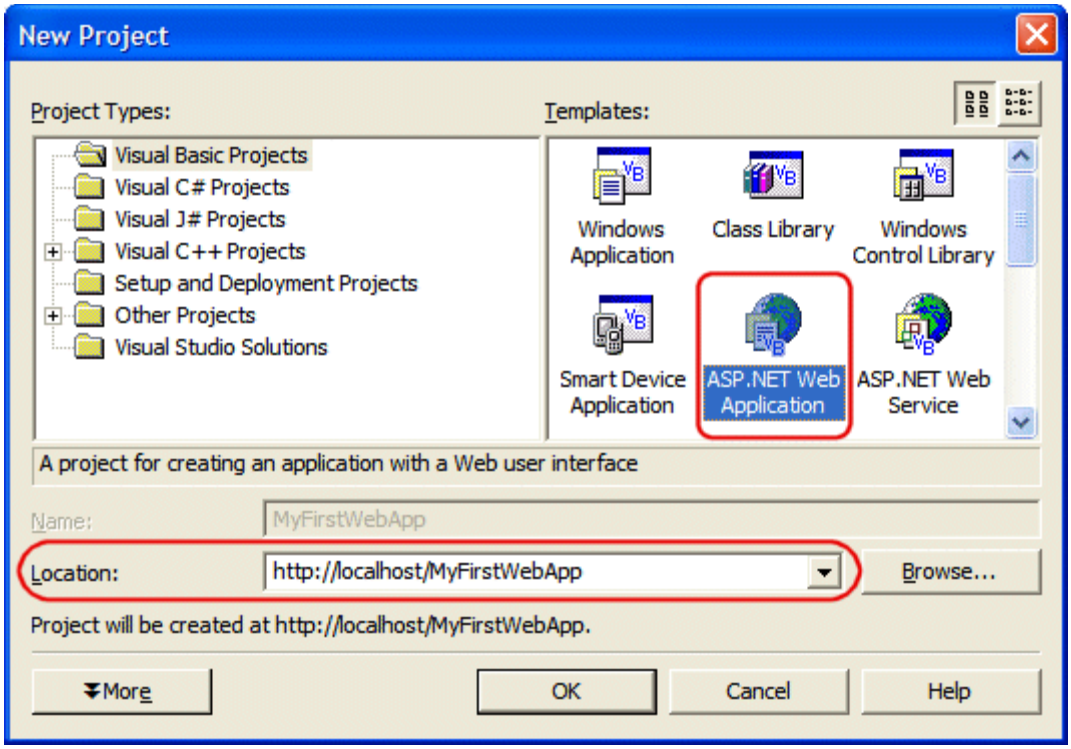
### *Steps to Create a Web Project*

1. Click **File > New > Project** to display the New Project dialog box.
2. In the Project Types pane, select either Visual Basic or Visual C#.
3. In the Templates pane, select **ASP.NET Web Application**.
4. In the **Location** text box, specify the URL where you want to place the Web project, including the name you want to give it.

   For example, to create a project titled **MyFirstWebApp** at the default localhost location, Insert the following URL string in the Location text box:

   ```
   http://localhost/MyFirstWebApp
   ```

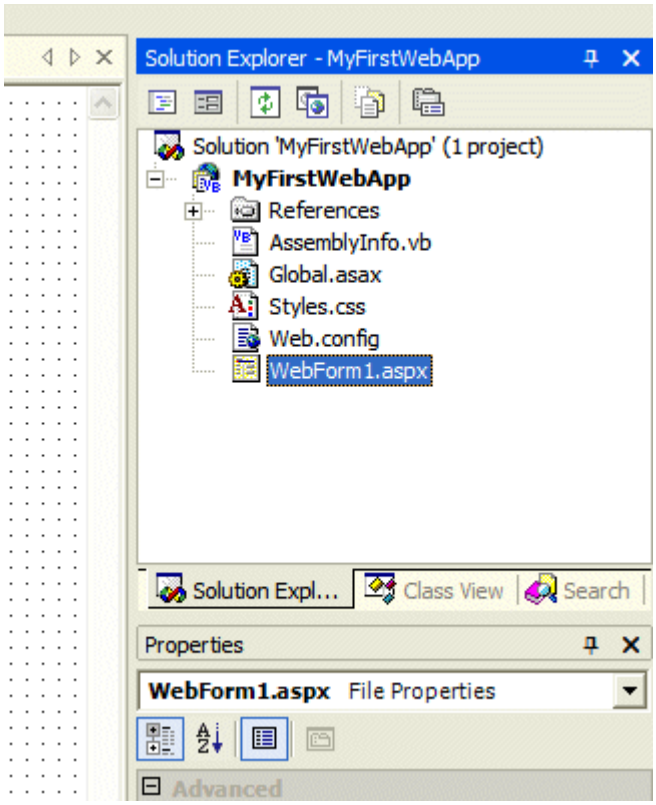
This allows for exact placement of your project files in a specific directory on a Web server.




**NOTE** If the local computer you are working on is also a Web server, you can create your Web project on that computer. The first time you start Visual Studio, the default location of a Web project is your local computer. In subsequent Visual Studio sessions, you can select the location of a previously used Web server from a drop-down list. If you want to choose a different server, you can click the Browse button to find other servers on your network, or you can type in the URL to a server (for example, http://domain.com) and press ENTER.

5. Click **OK** to create the Web project.

   Visual Studio connects to the server and adds the appropriate project items to Solution Explorer in the project node.




## Using Web Forms

Web Forms and ASP.NET were created to overcome some of the limitations of ASP. These new strengths include:

- Separation of HTML interface from application logic
- A rich set of server-side controls that can detect the browser and send out appropriate markup language such as HTML
- Less code to write due to the data binding capabilities of the new server-side .NET controls
- Event-based programming model that is familiar to Microsoft Visual Basic® programmers

- Allows third parties to create controls that provide additional functionality. For example the **Ektron CMS400.NET** Server Controls. *See Also:* "Introduction to Ektron CMS400.NET Server Controls" on page 24

On the surface, Web Forms seem just like a workspace where you draw controls. In reality, they can do a whole lot more. But normally you will just place any of the various controls onto the Web Form to create your UI. The controls you use determine which properties, events, and methods you will get for each control.

There is an excellent step by step explanation of creating and using Web Forms located on the following Web page:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dndotnet/html/introwebforms.asp

# Creating a Site Using Macromedia Dreamweaver MX 2004

**NOTE** The information in this section has been copied from the Macromedia Web site. It appears on the following Web page:

http://www.macromedia.com/cfusion/knowledgebase/index.cfm?id=tn_14567

The information has been modified by Ektron, Inc. to discuss .NET instead of Coldfusion and has been updated to reflect Macromedia Dreamweaver MX 2004 instead of Macromedia Dreamweaver 3.0

You can easily create a web site with Macromedia Dreamweaver MX 2004. This section discusses:

- creating a site
- creating files
- creating a template
- applying the template
- viewing the site

This chapter does not explain the use of **Ektron CMS400.NET** functions when building a Dreamweaver site. For Information on **Ektron CMS400.NET** functions in Dreamweaver, see "Dreamweaver Extension Support" on page 203.

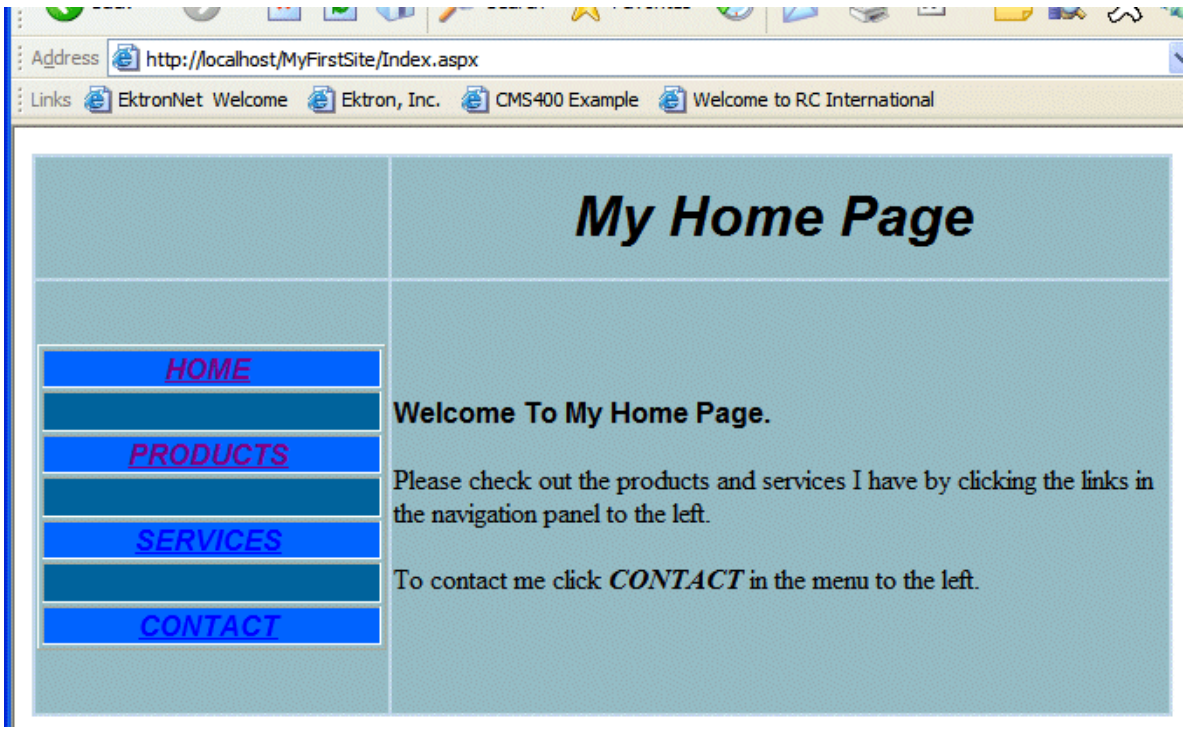
**IMPORTANT**

**Ektron CMS400.NET** Dreamweaver functions are an integral part of using **Ektron CMS400.NET** and Dreamweaver together. After completing this section, read "Dreamweaver Extension Support" on page 203.

The process of creating a basic site in Macromedia Dreamweaver MX 2004 can be broken down into the following six segments.


- "Defining a Local Site" on page 9
- "Building the Site Map in the Files Window" on page 12
- "Creating a Subfolder and Files" on page 17
- "Creating the Template" on page 17
- "Applying the Template" on page 23
- "Testing and Viewing the Site" on page 23


By completing the examples in these sections, you create a site with a template, hyperlinks and multiple pages. When you are completed, your site should look similar to the picture below. You might choose different colors and text but the layout should be the same.



## Defining a Local Site

Before creating pages for a site, it is necessary to define the site in Dreamweaver. This is done by selecting a folder on the computer's hard disk in which all the files for the site will be contained. This is the local root folder.

After the site is defined, the pages for the site are created in Dreamweaver, saved inside the local root folder, and then uploaded to the Web server. The local site and the remote site on the server should have exactly the same structure.

**NOTE** If Dreamweaver is used to upload the site, it automatically replicates the file structure of the local root folder on the server.

### *Steps to Define a Folder for Use as a Local Site*

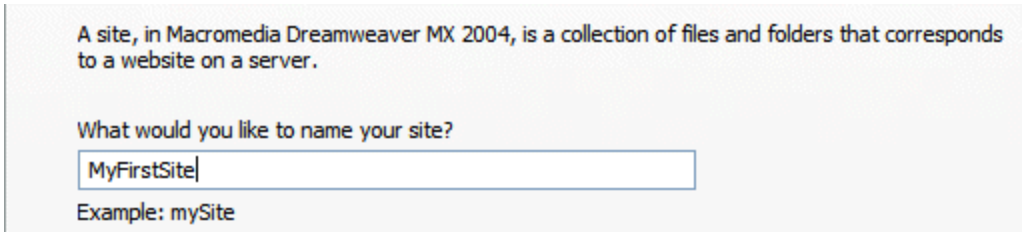
To define the folder as the local root folder in Dreamweaver, do the following:

1. On the welcome screen click Dreamweaver Site…

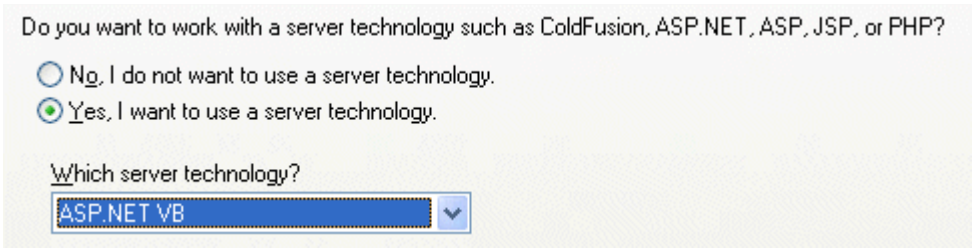
Or

Click **Site > Manage Site > New > Site**

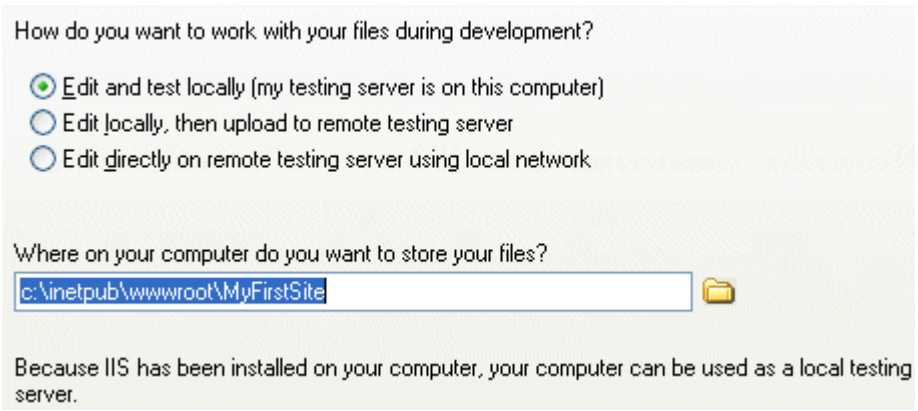
2. Enter the name of your site in the text box.


A site, in Macromedia Dreamweaver MX 2004, is a collection of files and folders that corresponds to a website on a server.
What would you like to name your site?
MyFirstSite
Example: mySite

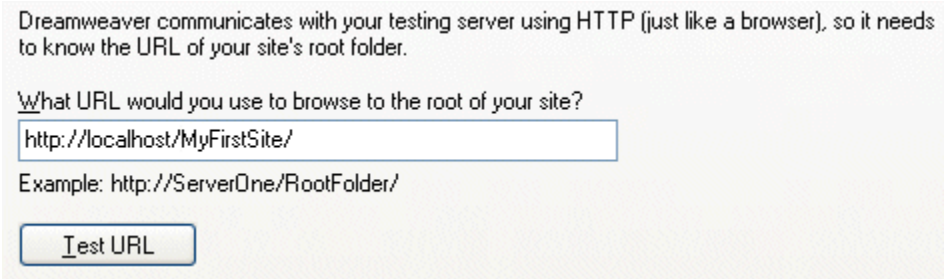
3. Click **Next**.
4. Click the **Yes, I want to use server technology** radio button.
5. Choose **ASP.NET** from the **which server technology** dropdown box.


Do you want to work with a server technology such as ColdFusion, ASP.NET, ASP, JSP, or PHP?
No, I do not want to use a server technology.
Yes, I want to use a server technology.
Which server technology?
ASP.NET VB

6. Click **Next**.
7. Choose how you want to work with your files during development. For these example steps choose the **Edit and test locally (my testing server is on this computer)** radio button.
8. Choose where you want to store the files.


How do you want to work with your files during development?
Edit and test locally (my testing server is on this computer)
Edit locally, then upload to remote testing server
Edit directly on remote testing server using local network
Where on your computer do you want to store your files?
c:\inetpub\wwwroot\MyFirstSite
Because IIS has been installed on your computer, your computer can be used as a local testing server.

9. Click **Next**.
10. Add the URL to browse to the root of your site.

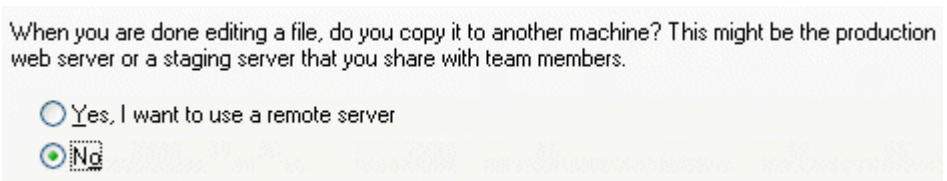

Dreamweaver communicates with your testing server using HTTP (just like a browser), so it needs to know the URL of your site's root folder.
What URL would you use to browse to the root of your site?
http://localhost/MyFirstSite/
Example: http://ServerOne/RootFolder/
Test URL

11. Click **Test URL**. If everything works the following message appears.

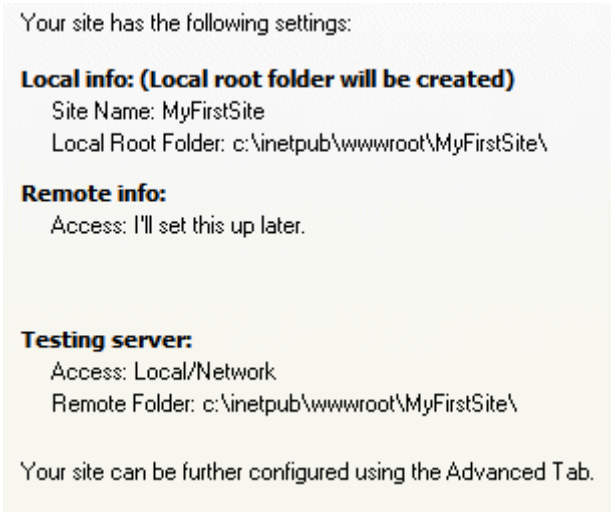
If this message does not display, an error message displays to help direct you on how to fix the issue. For example:

Dreamweaver MX 2004
Dreamweaver cannot use the prefix you entered to display live data. Please double-check your site configuration or click on Help for more information on how to correct this problem. (HTTP Error: 404)
Don't show me this message again.
OK

12. Click **Next**.
13. Choose if you want to copy your site to a remote server. For these example steps choose the **No** radio button.


When you are done editing a file, do you copy it to another machine? This might be the production web server or a staging server that you share with team members.
Yes, I want to use a remote server
No

14. Click **Next**.
15. A summary appears.


Your site has the following settings:
Local info: (Local root folder will be created)
Site Name: MyFirstSite
Local Root Folder: c:\inetpub\wwwroot\MyFirstSite\
Remote info:
Access: I'll set this up later.
Testing server:
Access: Local/Network
Remote Folder: c:\inetpub\wwwroot\MyFirstSite\
Your site can be further configured using the Advanced Tab.

**NOTE** If you feel comfortable using Dreamweaver to create your Web site, you can click the Advanced Tab and further define your Web site settings. You can access help for the Advance Tab in Dreamweaver.

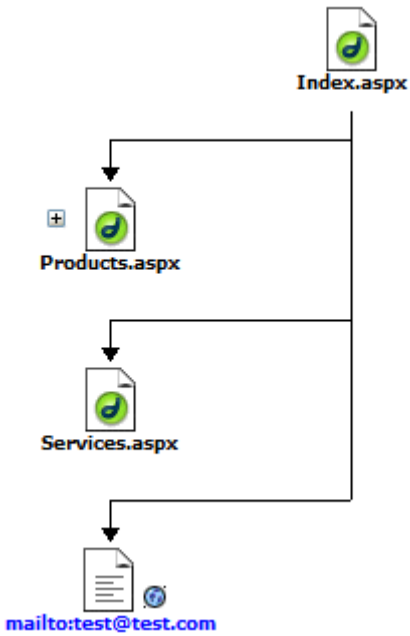
16. Click **Done**.

## Building the Site Map in the Files Window

Now that the site has been defined, the Files window appears on the right side of the Dreamweaver window. The Files window provides two visual representations of the local site: Site Map and Site Files.

The Files window can be used to add new files to the site, and to add, remove, and change links. It can also be used to create an

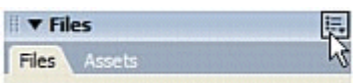
image file of the site that can be exported to and printed from an image editing application.




This is useful for creating a Web page with a visual layout of your site and linking each page to the image. This is also useful if you preparing a report on your site and would like to have a site map image included in the report.

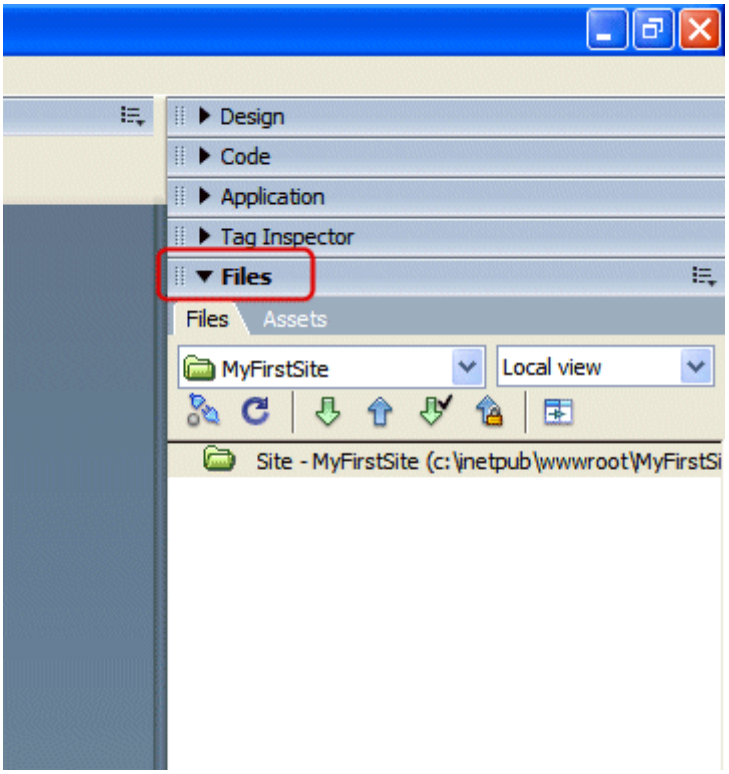
To create a site map image, follow these steps:

1. In the **Files window**, click **Map view** from the dropdown box on the right.
2. Click the dropdown Menu as shown below.




3. Click **File > Save Site Map**.
4. In the File Type dropdown box, select .bmp or .png.
5. Enter a name in the **File Name** text box.
6. Select a location to save the file.
7. Click **Save**.

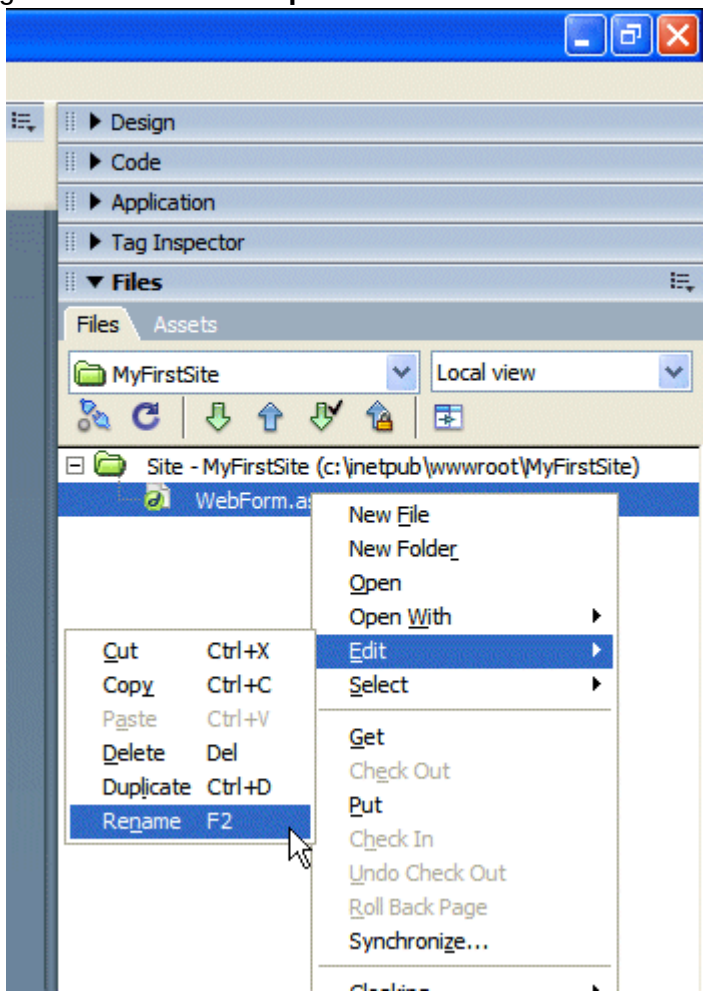
A home page needs to be created and defined in order to build the site map.




### *Steps to Add a Home Page*

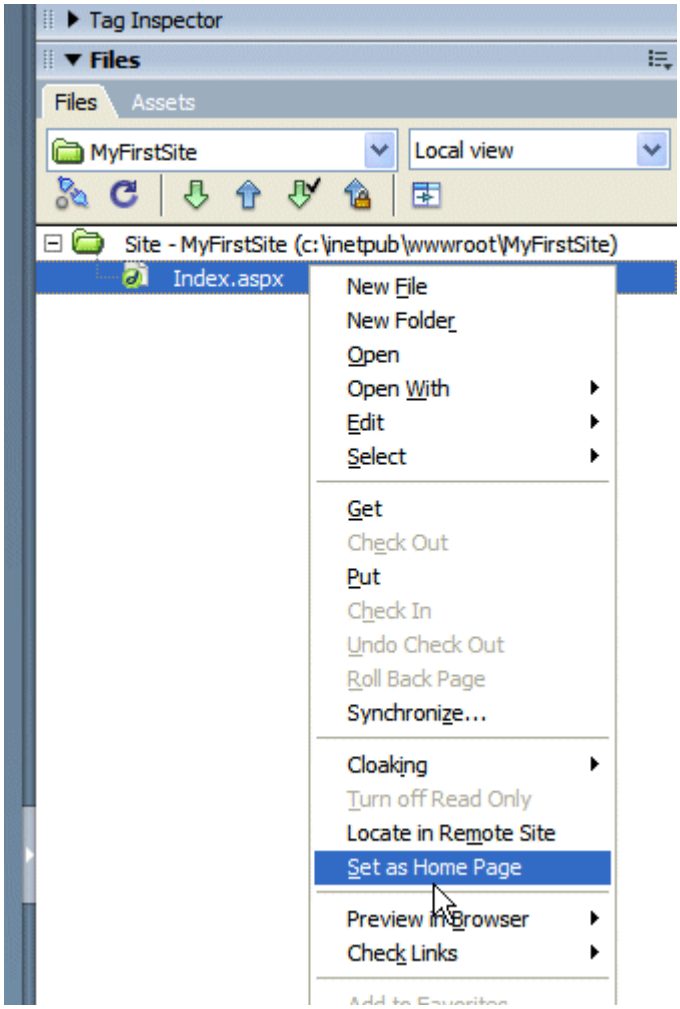
1. Right click the local root folder in the Files window. For example, Site - MyFirstSite (c:\inetpub\wwwroot\MyFirstSite).
2. Click **New File**.
3. A file titled **WebForm.aspx** appears in the Files window.

4. Right click **WebForm.aspx** and choose **Edit > Rename**.

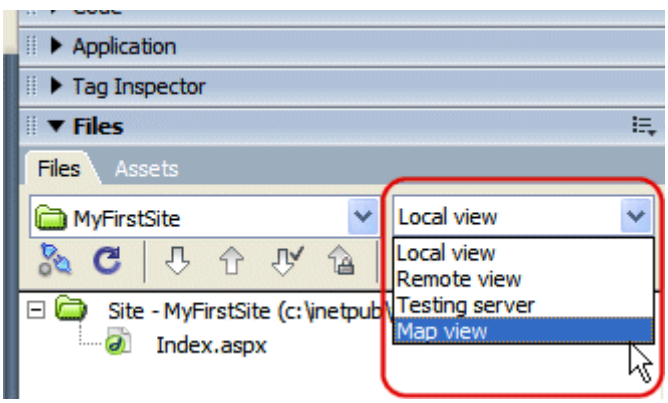



5. Name the file Index.aspx.

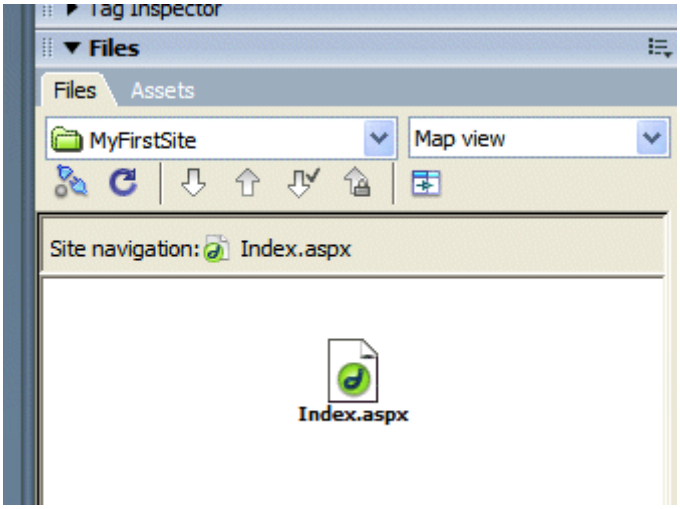
6. Right click **Index.aspx** and select **Set as Home Page**.




7. To view a site map, click **Map view** from the dropdown menu in the Files window.



8. The Map view appears.




## Creating a Subfolder and Files

The home page currently has no links. After links have been established, the home page and the linked pages will be displayed in the site map.

### *Steps to Create a Subfolder and Files*

1. If you are in Map view, switch back to **Local view**.
2. Right click the local root folder in the Files window. For example, Site - MyFirstSite (c:\inetpub\wwwroot\MyFirstSite).
3. Click **New Folder**.
4. Name the new folder **Pages**.
5. Create two files to put into the **Pages subfolder** by right clicking the subfolder and choosing **New File**.
6. Rename the two files Products.aspx and Services.aspx.

**NOTE** The template will be applied to these files in the following steps.

## Creating the Template

Templates allow a designer to construct a site that has a consistent style and is easy to update. Templates are convenient for a team of designers, allowing some members of the team to concentrate on

the style of the pages, while other members of the team create new pages and revise the content without altering the formatted layout. Changes can be made to the overall design of the site quickly by revising the template files directly.

### *Steps to Create a Template*

The Create a Template example is broken into three parts:


- ”Creating a New Template” on page 18
- ”Creating a Navigation Bar for the Template” on page 19
- ”Establishing Editable Regions and Saving the Template” on page 21


#### Creating a New Template

1. Click **File > New...**
2. On the General Tab, click **Template page > ASP.NET VB template**.
3. Click **Create**.
4. Choose **Modify > Page Properties**. In this dialog box you can do the following:

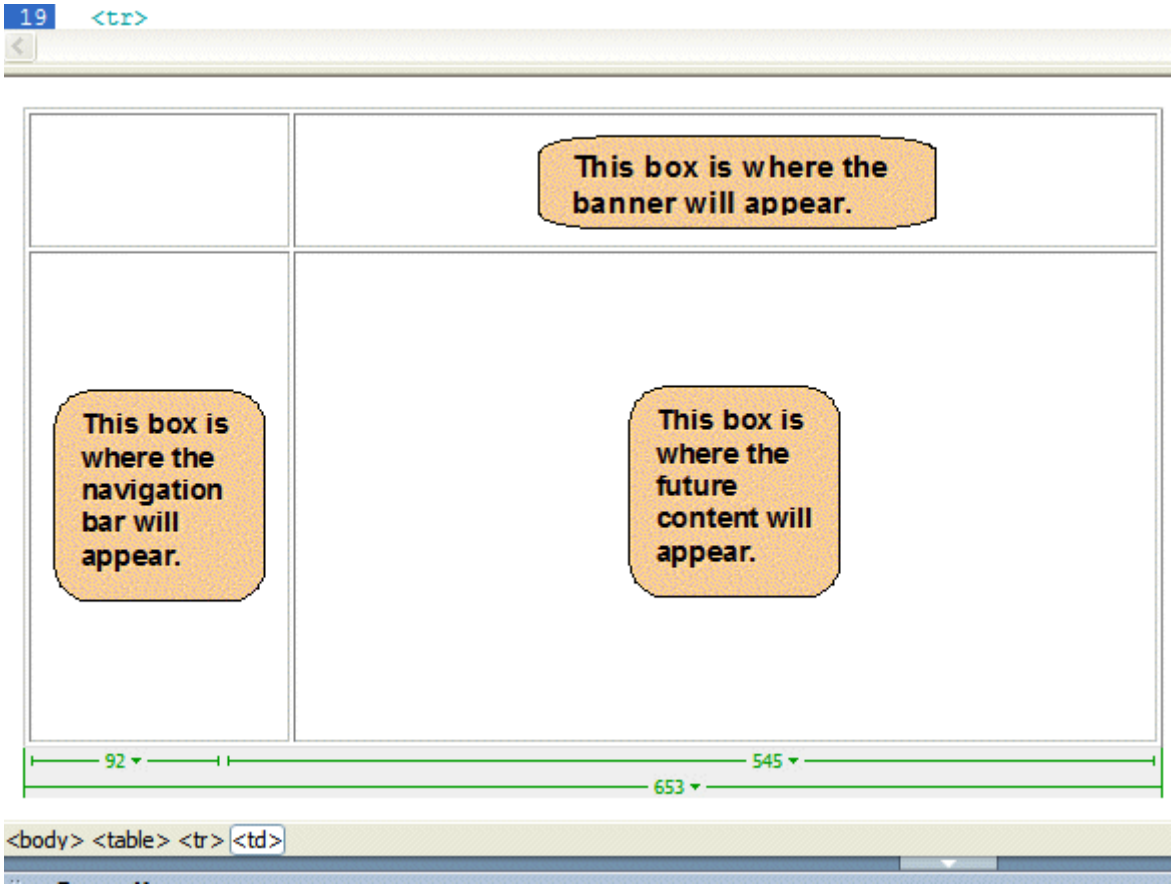
- choose a title for the template
- set the template's background
- set text
- choose links colors
- set left and top margin, and margin width and height.

5. Insert a two row, two column table in the template.

- The first row is reserved for the page banner and will be an editable region.
- The second row of the first column is where the navigation bar will be built. This row will be an uneditable region.

- The second row of the second column is where the future content will be placed.

  It should look like this




**Creating a Navigation Bar for the Template**

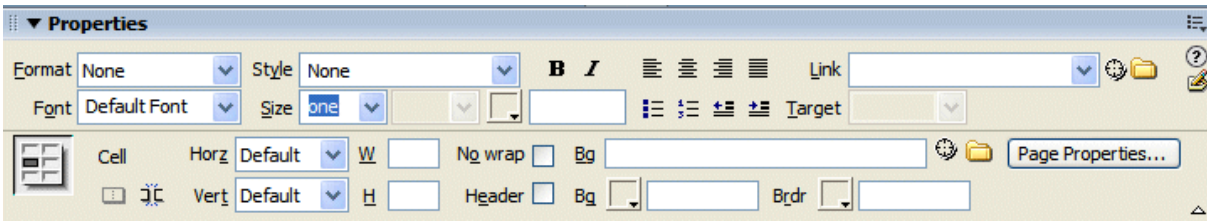
Most Web sites have a navigation bar that provides links to the individual Web pages of the site. Usually, the navigation bar is inserted into the uneditable region of the template. This method ensures it is easily applied to each page created from the template and is updated by simply editing the template.

1. In the second row of the first column, insert a one column, seven row table.
2. In the first row of this embedded table, type **HOME**.
3. In the third row, type **PRODUCTS**.

4. In the fifth row, type **SERVICES**.
5. In the seventh row, type **CONTACT**.

   Your page should look like this:

6. With the cursor in a cell, the Property inspector displays the cell properties, such as background color (Bg), width (W), height (H), and text alignment (Horz/Vert). Apply the desired properties to each cell.



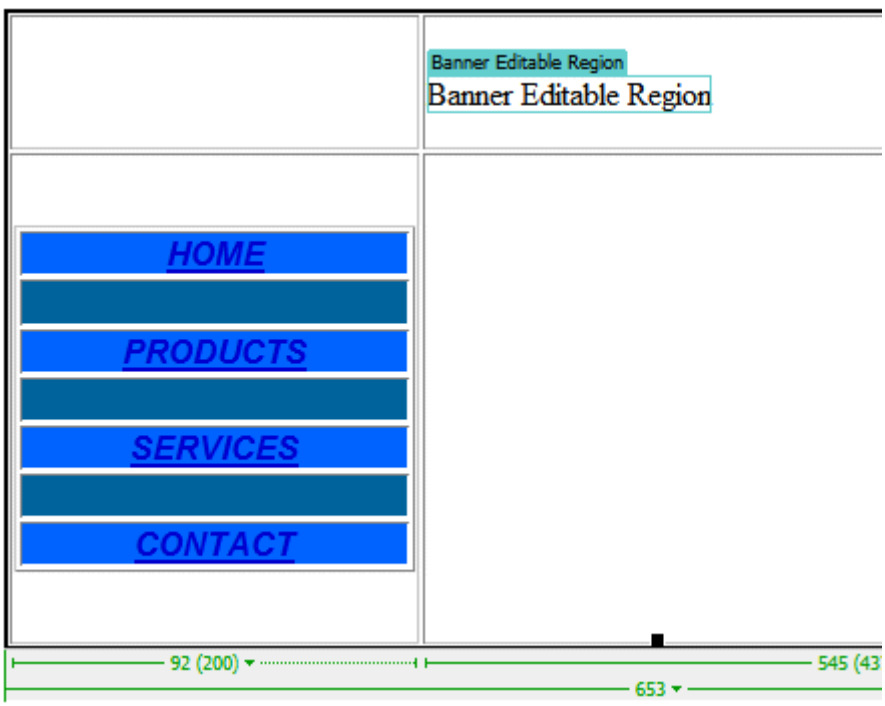

7. To link to index.aspx, products.aspx, and services.aspx, select the text in the cell, click the Folder icon in the Link field of the Property inspector, browse to the appropriate file, and select the file. Repeat this process for each cell.
8. For the Contact link, select the CONTACT text and, in the Link field of the Property inspector, type in **mailto:youremailaddress**. When a visitor to the site selects that link, their e-mail program opens and an e-mail with your address in the To: field becomes available.

### Establishing Editable Regions and Saving the Template

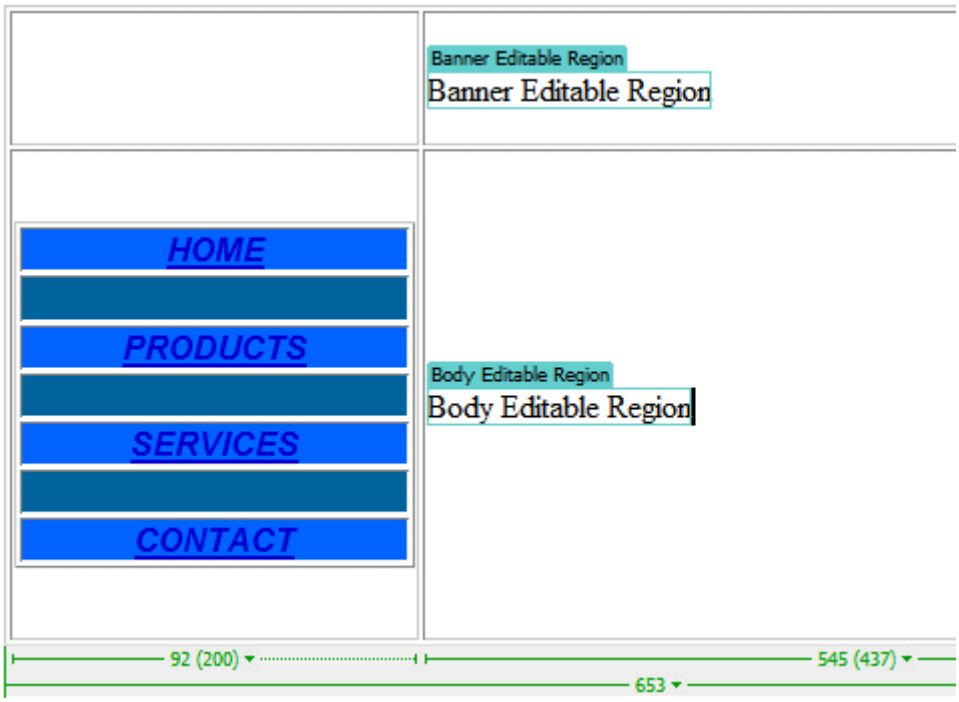
Once the design of the future template is complete, it must be saved as a DWT file (template) in order to be able to establish editable or noneditable regions.

Originally, each template has only locked regions. Locked regions can be edited only inside the template itself. Editable regions are placeholders for the content that is unique to each page to which the template is applied. The content of editable regions is created and edited in the document that is derived from the template.

1. With the cursor in the first row, second column, select **Modify > Templates > New Editable Region** or click the **Editable Region** button ( ).
2. The New Editable Region dialog box appears.
3. In the **Name** field, type **Banner Editable Region** as the name for this region of the template. This editable region will be used as a placeholder for the title of the site pages.
4. Click **OK**. In the Document window, the region is highlighted and identifiable by the region name.



5. With the cursor in the empty cell next to the navigation bar (the second row of the second column), **Modify > Templates > New Editable Region** or click the **Editable Region** button ( ).
6. In the **Name** field, type **Body Editable Region** as the name for this region of the template. This editable region will be used as a placeholder for the text of the site pages.
7. Click **OK**. In the Document window, the region is highlighted and identifiable by the region name.




8. Click **File > Save**.
9. In the **Save as:** text box choose a name.
10. Click **Save**.

**NOTE** Make sure not to move the DWT file from the Templates folder. Also, do not rename or move the Templates folder. As long as the DWT file remains inside the Templates folder, the links from the pages derived from the template to the template will not be broken.

## Applying the Template

The new template is ready to apply to the index, products and services pages. In an applied template, the editable regions are highlighted. They appear in a highlight color on all pages to which the template is applied as well.

### *Steps to Apply the Template*

1. Select **File > Open**.
2. Browse to and select **Index.aspx**.
3. Select **Modify > Templates > Apply Template to Page**.
4. Click the **Template**.
5. Click **Select**.
6. In the table, type the title of the page in the **Banner Editable Region**. Type or insert text in the **Body Editable Region**.
7. Click **File > Save**.
8. Repeat these steps for the Products.aspx and Services.aspx pages.

## Testing and Viewing the Site

You are now ready to test and view the site.

### *Steps to Test and View the Site*

1. In the Document window, select **File > Open**.
2. Browse to and select **Index.aspx**.
3. Select **F12** to preview in the browser.
4. Click the navigation bar links to navigate from page to page.

# Introduction to Ektron CMS400.NET Server Controls

**Ektron CMS400.NET**'s server controls let you insert many standard methods and properties within the Visual Studio .NET environment. This means that you can see the effect of your changes in real time -- you don't have to modify a page then compile a sample project to see the results.

You can insert server controls using drag and drop or programmatically. You can also use databinding to retrieve and display data from **Ektron CMS400.NET**. This chapter explains the use of the Ektron CMS400 Server Controls through the following topics.


- "Opening the Sample Project" on page 25
- "Setting Up a Template" on page 26
- "Making the Server Controls Available" on page 27
- "Inserting Server Controls Using Drag and Drop" on page 29
- "Inserting Server Controls Programmatically" on page 33
- "Using Drag & Drop and Programmatically Together" on page 51
- "Data Binding with Server Controls" on page 53
- "Caching with Server Controls" on page 59
- "List of Server Controls" on page 61


Additional information and examples for Ektron Server Controls are available online by going to the Developers section of the sample site that was installed. Here is the path:

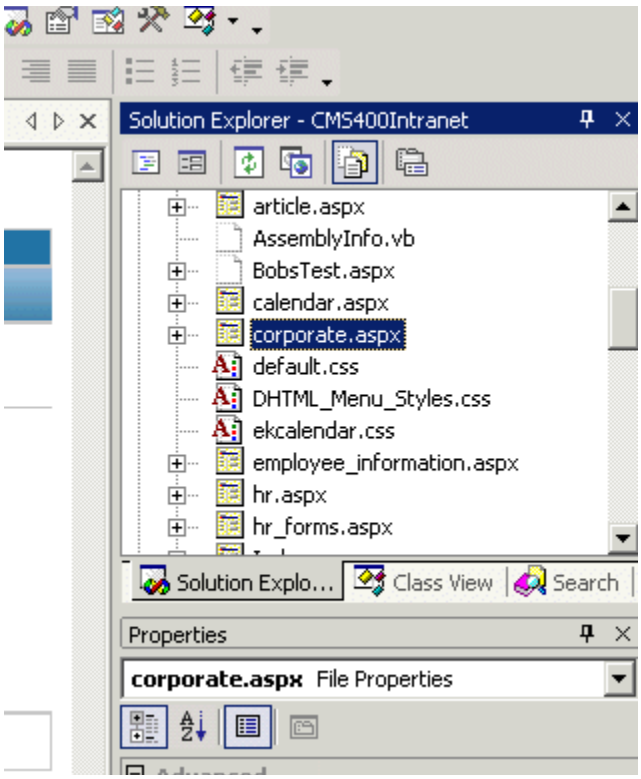
- `http://localhost/CMS400Example/Developer.aspx`

Replace `localhost` with the `webroot` where you installed the Example site.

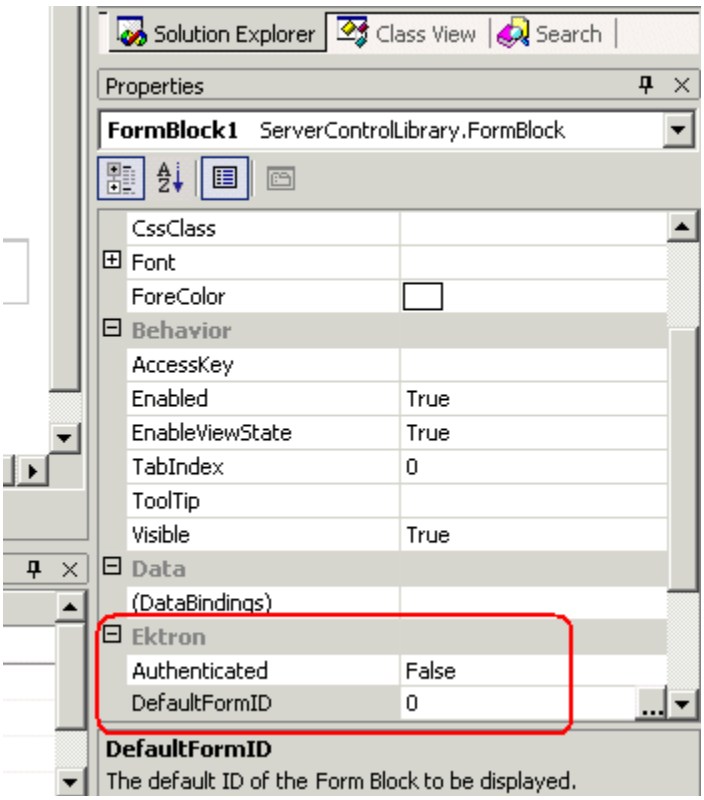
# Opening the Sample Project

To help get you started, these directions explain how to open the sample site delivered with **Ektron CMS400.NET**.

1. Browse to and double click Ektron CMS400.NET's solution file, `localhost/CMS400Example/CMS400Example.sln`.
2. The sample site project opens.
3. To work on a template page, click it from the Solution Explorer.



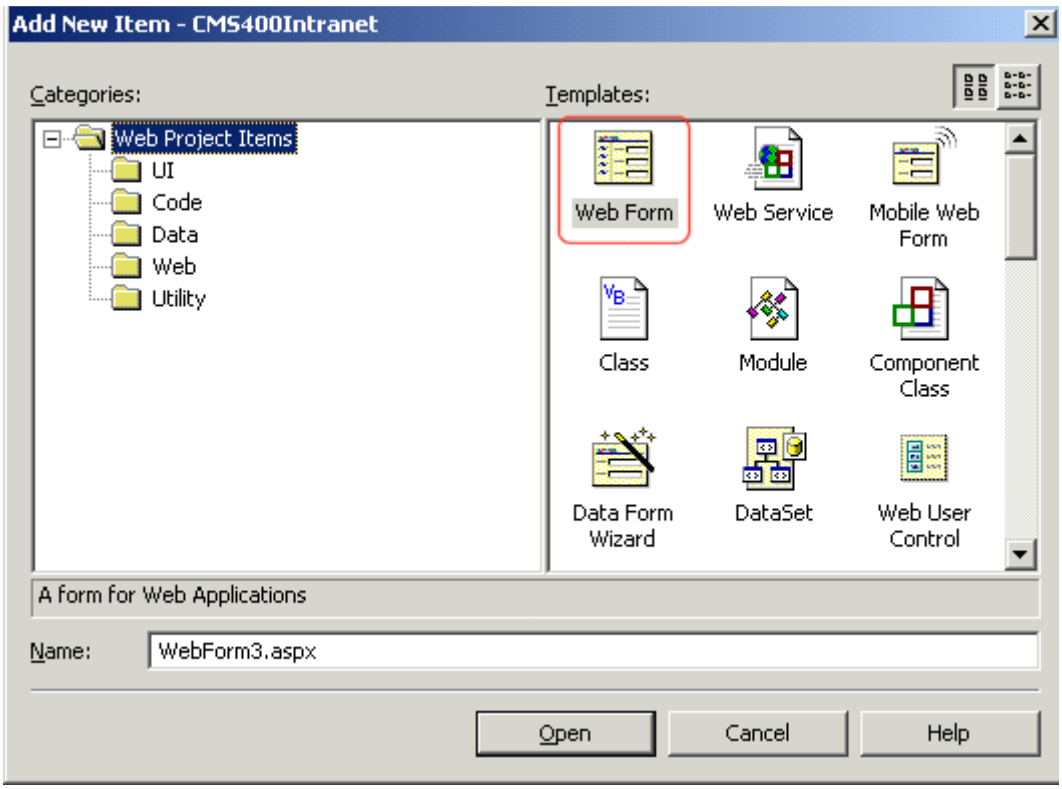

When you select a page, its properties appear in the Properties area (below Solution Explorer), and the page appears in the center of the screen. A control's properties include several standard .NET properties along with Ektron-specific ones. The Ektron properties are labeled as illustrated below.




# Setting Up a Template

To create a new template (.aspx) page, follow these steps.

1. Click **Project > Add Web Form**.
2. On the Add New Item screen, click Web Form and assign a name.



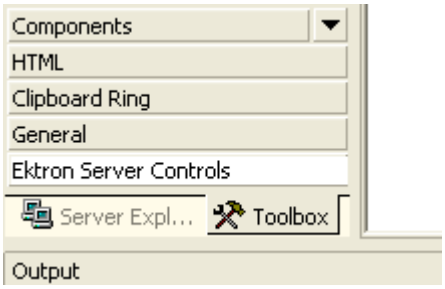

3. Add controls to determine the page content.

# Making the Server Controls Available

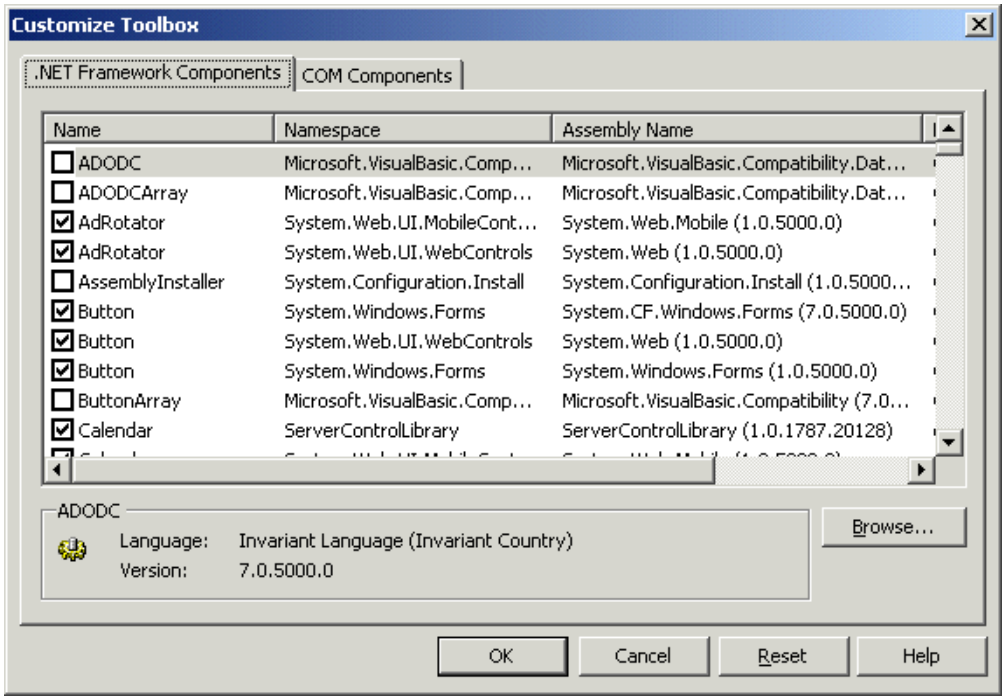
**NOTE** You must copy the dlls to a local drive before installing them. You cannot copy them from a network drive.

1. Display the Visual Studio .NET toolbox (**View > Toolbox**).
2. Right click the mouse within the Toolbox.
3. Click **Add Tab**.

4. Type **Ektron Server Controls** then press enter.




5. Click the Ektron Server Controls Tab.
6. Right click the mouse in the empty area.
7. Click **Add/Remove Items**.
8. The Customize Toolbox dialog appears.




9. Select the .**NET Framework Components** tab.
10. Browse to the directory that stores Ektron CMS400.NET's dll files, `localhost/CMS400Example/bin`.

11. Add the Ektron.Cms.Controls.dll file and press **OK**. This file provides access to **Ektron CMS400.NET**'s server controls.

### Removing the Server Controls

1. Display the Visual Studio .NET toolbox (**View > Toolbox**).
2. Right click the mouse within the Toolbox.
3. Click **Add/Remove Items**.
4. Click **Namespace** or **Assembly Name** to sort the Server Controls by manufacturer.
5. Uncheck all boxes that are Ektron.Cms.Controls.
6. Click **OK**
7. Right click the Ektron Server Control Tab.
8. Click **Delete Tab** to remove it.

### Updating the Server Controls

To update the server controls you must first remove the existing server controls in Visual Basic, then add the new server controls.

For information on removing server controls, see "Removing the Server Controls" on page 29

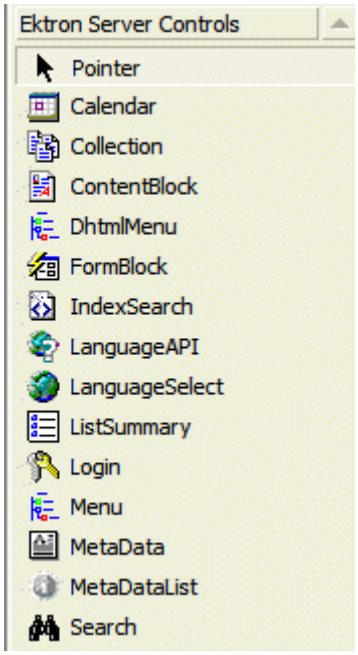
For information on adding server controls, see "Making the Server Controls Available" on page 27

## Inserting Server Controls Using Drag and Drop

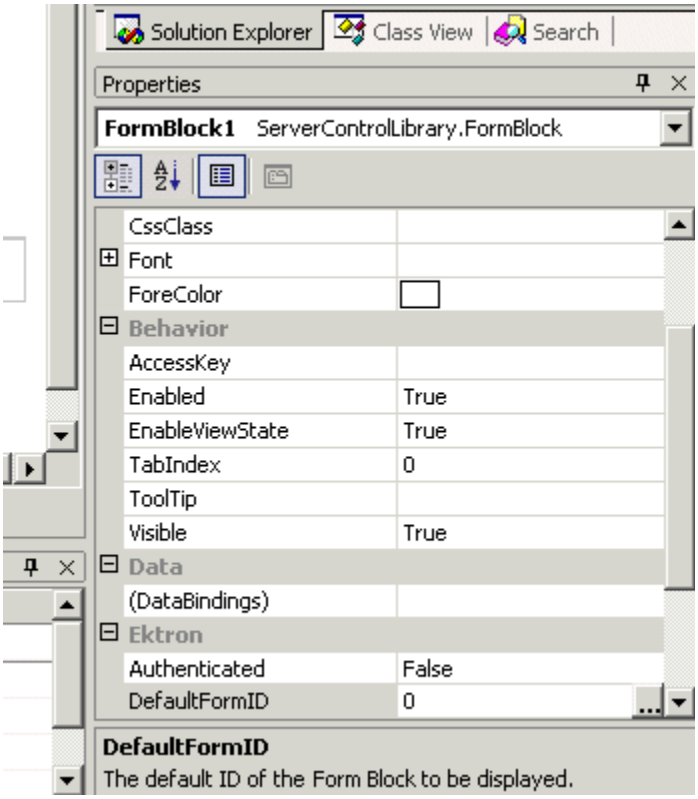
Because Visual Studio .NET is a visual environment, you can watch the page layout change as you add or move a control and adjust its properties. And, whenever you need to identify an **Ektron CMS400.NET** object (such as content block or collection), you can use the CMS Explorer to visually browse to and select it.

### Adding a Server Control to a Template

1. Display the Visual Studio .NET toolbox (**View > Toolbox**).
2. Click the **Ektron Server Control** tab. **Ektron CMS400.NET**'s server controls appear.




3. Drag a server control and drop it into the desired location on the page.
4. As desired, modify the control's properties using the Properties area of Visual Studio .NET.




For a description of server control properties, see "List of Server Controls" on page 61.

### *ServerControlWS.asmx*

> Best Practice
>
> For improved security, Ektron recommends renaming or removing the Web services file when you move it to your production server. After installation, this file is named ServerControlWS.asmx and resides in the /CMS400Example/Workarea/ folder in your Web root.

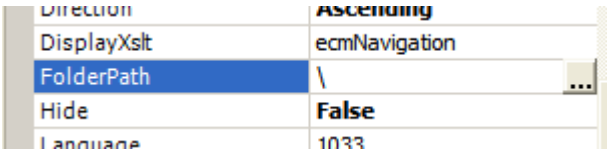
ServerControlWS.asmx is the Web service that lets the server controls talk to **Ektron CMS400.NET**. The path is coded in the web.config file. This is how it appears:

```
<!-- Web Service URL for Server Controls design time  --> <add key="WSPath"
value="http://localhost/cms400example/Workarea/ServerControlWS.asmx" />
```

You will need to edit this line if you change the location or name of the ServerControlWS.asmx file.

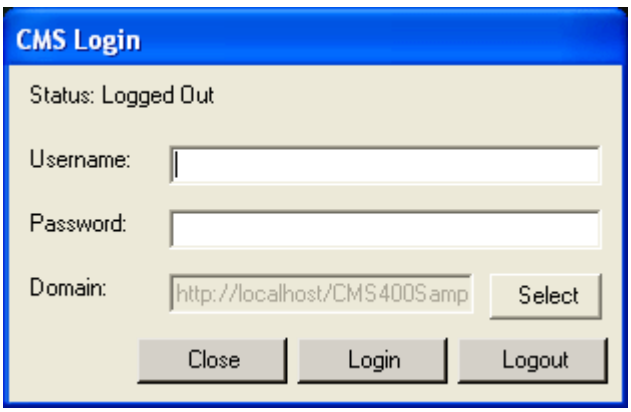
### *Using CMS Explorer to Browse Your Ektron CMS400.NET Site*

**Ektron CMS400.NET** provides a window (called the CMS Explorer) that lets you browse through your Web site to identify CMS objects such as folders, calendars and content blocks. For example, if you insert a List Summary Server Control, its **FolderPath** property identifies the folder whose contents are displayed. If you are not sure of the folder path, click the gray button.

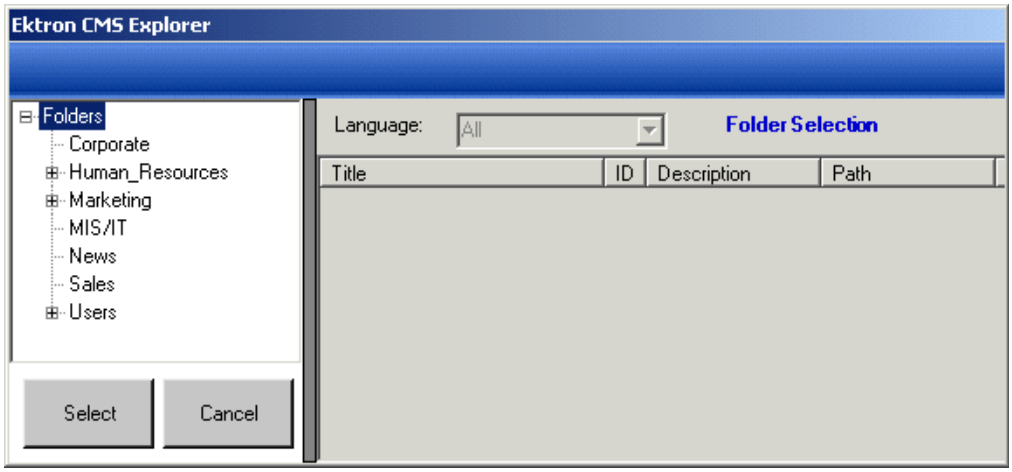



When you do, a login screen appears (illustrated below).

**NOTE** A user cannot be logged into the CMS Explorer and the CMS400.NET Workarea at the same time. If you log into the CMS Explorer while logged into the Workarea, you will receive the following error when you try to use the Workarea: `Error: The current user does not have permission to carry out this request [CBR R31].` If you log into the Workarea while logged into the CMS Explorer, you will need to re-login to the CMS Explorer when you return to using it.



Enter your **Ektron CMS400.NET** username and password. When you do, the CMS Explorer window appears, providing a visual way to browse your **Ektron CMS400.NET** Web site.




Navigate through the folders, select an object, then click the **Select** button. The selected object is pasted to the server control property.

**NOTE** Although you see the object in the selected language in Visual Studio .NET, the language is not stored. For example, if you select a German content block whose ID=2, **Ektron CMS400.NET** only stores `content block ID=2`. When a visitor to your site browses to that page, the content block does not appear in the selected language. Instead, it is determined by a cookie or the user's language selection.

### Build the Project

After you open the solution file in Visual Studio .NET and add the required files, build the project. This may take a while the first time you do so. After the project is built, a browser opens and it is displayed as a Web page.

You can also view a Web page while working on it by right clicking on the Web form and clicking **View in Browser**.

## Inserting Server Controls Programmatically

Besides using the drag-and-drop method to insert a server control (explained in "Inserting Server Controls Using Drag and Drop" on

page 29), you can insert one programmatically. You might prefer to insert server controls programmatically for several reasons, such as

- you want the control to be loaded into memory only under certain conditions. In this case, insert the logic that only displays the control if the condition exists.
- you want to display only certain properties of an object, such as the title of last edited date of a content block.

Note that you can drag and drop controls on your page then customize them using code behind. For more information, see:"Using Drag & Drop and Programmatically Together" on page 51.

The following sections explain how to use the server controls programmatically.


- "Steps to Insert a Server Control Programmatically" on page 34
- "Server Control Properties in Code Behind" on page 38
- "Customizing Server Controls" on page 48


## Steps to Insert a Server Control Programmatically

To insert an Ektron server control programmatically, follow these steps.


"Step 1: Declare the Namespace" on page 35

"Step 2: Create an Instance of the New Control" on page 35

"Step 3: Set the Properties" on page 36

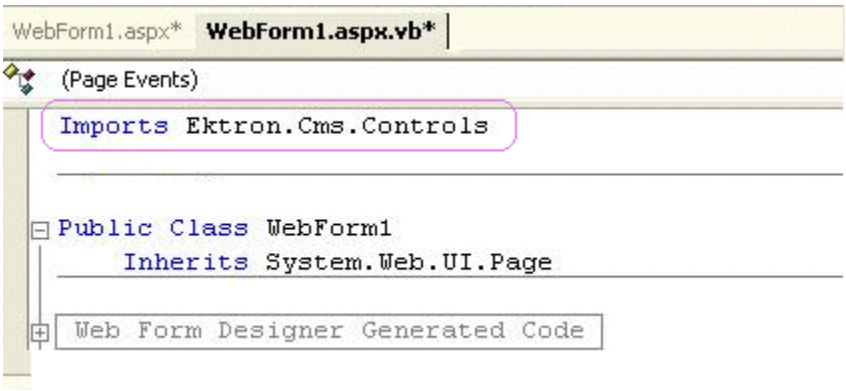
"Step 4: Call the Fill Method" on page 37

"Step 5: Use .text Property to Display the Control" on page 37


Each step is described below. These steps show an example of using the Collection Server Control.

### *Step 1: Declare the Namespace*

Before inserting the server control, you can declare the namespace at the top of the code behind Visual Basic file, as shown below.




**NOTE** You do not need to declare a namespace. However if you do not, you must fully qualify objects that you create.

> Best Practice
>
> For any customization of the CMS, classes or controls that inherit from Ektron classes, it is recommend that you create your own namespace within 'Ektron.Cms.Custom'. For example, if your company is 'AcmeExampleTech, Inc.' you should create all of your custom classes within the namespace 'Ektron.Cms.Custom.AcmeExampleTech'.

### *Step 2: Create an Instance of the New Control*

Next, declare a control as an object in the code behind. In this example, we create a collection named `MyColl`.

```
dim MyColl as New Collection
```

or

```
dim MyColl as new Ektron.Cms.Controls.Collection
```

With C#, use this syntax.

```
Ektron.Cms.Controls.Collection MyColl = new

Ektron.Cms.Controls.Collection();
```

You can declare any server control as an object by using the server control name in the "List of Server Controls" on page 61. Another example would be: `Dim MyMdl as New MetaDataList.`

### *Step 3: Set the Properties*

**NOTE** For descriptions of the properties and how to use them, see "Server Control Properties in Code Behind" on page 38.

After inserting the control, you can set the properties that you want to display on the page. For example:

```
Dim MyColl as New Collection

MyColl.DefaultCollectionID = 1
```

or

```
dim MyColl as new Ektron.Cms.Controls.Collection

MyColl.DefaultCollectionID = 1
```

With C#, use this syntax.

```
Ektron.Cms.Controls.Collection MyColl = new

Ektron.Cms.Controls.Collection();

MyColl.DefaultCollectionID = 1;
```

This line tells the page to display CollectionID 1 unless otherwise specified.

**NOTE** If you do not know an object's ID number, you can switch to Design mode, drag and drop the object, then use the CMS Explorer to find the ID number. (See "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32.) If you do this, remember to delete the dropped object when you are done.

You can also obtain the ID number via the Workarea.

```
Dim MyColl as New Collection

MyColl.DefaultCollectionID = 1

  MyColl.Random = True
```

This line sets the Random property to **true**. *See Also:* "Random" on page 117.

### *Step 4: Call the Fill Method*

Because there is no render event when using objects as components not as controls, use the Fill method to fill an object's properties on the page. An example is below.

```
Dim MyColl as New Collection

MyColl.DefaultCollectionID = 1

MyColl.Random = True

  MyColl.Fill()
```

### *Step 5: Use .text Property to Display the Control*

Finally, use the `.text` property to determine what appears on the Web page. For example, to display the first item in a collection, use this syntax.

**NOTE** Before adding this line you need to drag and drop a label on your Web form.

```
Dim MyColl as New Collection

MyColl.DefaultCollectionID = 1

MyColl.Random = True

MyColl.Fill()

Label1.Text = myColl.EkItems(0).Title
```

To display *all* items in a collection, use this syntax.

```
  Dim myColl As New Collection
  Dim ekitem As New Ektron.Cms.Common.ContentBase
  myColl.DefaultCollectionID = 2
  myColl.Fill()
Label1.Text = "<ul>"

For Each ekitem In myColl.EkItems

    Label1.Text &= "<li><a href=""" & ekitem.QuickLink & """>" & ekitem.Title & "</a>"

Next

Label1.Text &= "</ul>"
```

This example displays the quick link for every content block in the collection, formatted as a bulleted list. You can use similar code to display a List Summary or search results. The following explains the new (red) code above.

- `label1.Text = "<ul>"` displays the opening tag for the bulleted list
- `For Each ekitem In myColl.EkItems` creates a loop for all content blocks in the collection
- `label1.Text &= "<li><a href=" & ekitem.QuickLink & ">" & ekitem.Title & "</a></li>"` for each content block in the collection, displays its quicklink and title
- `Next` loops through all content blocks in the collection
- `label1.Text &= "</ul>"` closes the bulleted list

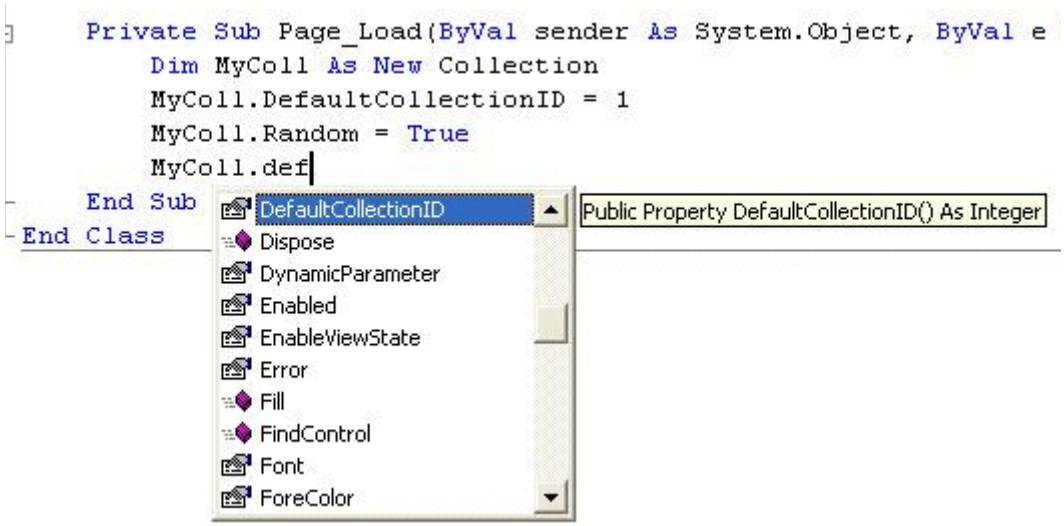
## Server Control Properties in Code Behind

Every server control has properties associated with it that you can only access programmatically. This subsection explains, what they are, how to access them, and how to use them.


- ”Displaying Properties for a Control” on page 38
- ”Mutual Server Control Properties in Code Behind” on page 39
- ”Accessing Additional Properties” on page 41
- ”Accessing Items in an Array” on page 47
- ”Referencing the Page Property” on page 47


### *Displaying Properties for a Control*

You can use Visual Studio .NET’s intellisense feature to display a control’s properties. The intellisense box appears as soon as you insert the period (.) after the object, as illustrated below.




The intellisense box displays *all* properties that can be applied. For a description of Ektron's standard server control properties, see "List of Server Controls" on page 61. To learn about native VS .NET properties, see its documentation.

The property's tooltip text indicates its type. In the above example, you can see that the DefaultCollectionID's type is integer.

For more information about accessing **Ektron CMS400.NET** object properties, see "Customizing the Server Control in the Code Behind" on page 50.

### *Mutual Server Control Properties in Code Behind*

There are several read-only properties that can be called in the code behind and used by every server control. The list below explains these properties.

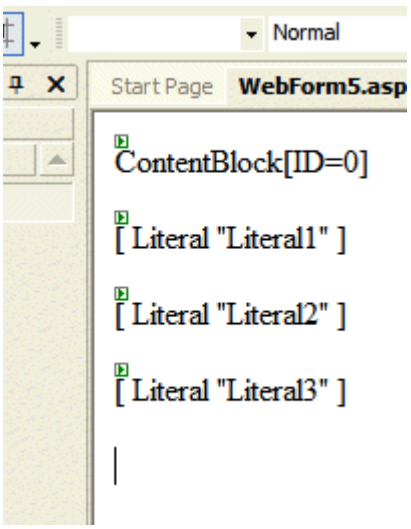
**NOTE** The properties do not display values within Visual Studio.NET during design time. Instead, they only display values at run time, which are dependent on the user's login status.

| Property | Value | Data Type |
|---|---|---|
| IsLoggedIn | Tells if a user is logged in to **Ektron CMS400.NET**.<br>**True** = User is logged in<br>**False** = User is not logged in | Boolean |
| LoggedInUserName | Gets the **Ektron CMS400.NET** user name to display. | String |
| LoggedInUserID | Gets the **Ektron CMS400.NET** ID of the user to display. | Integer |

These properties will allow you to personalize any page with your users names and IDs, and show if they are logged in. Here is an example of using these properties in code behind.

**NOTE** You must be logged in to **Ektron CMS400.NET** for this example to show your name and ID.

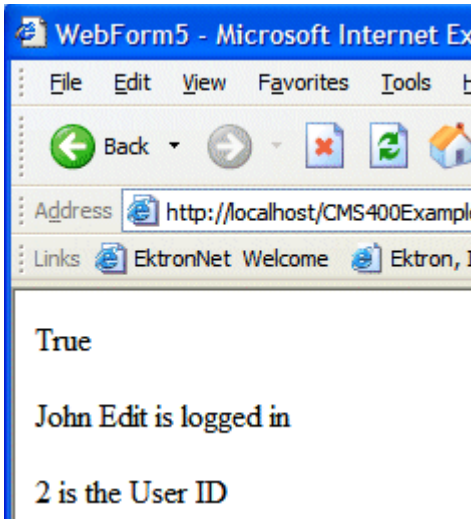
1. Drag an **Ektron CMS400.NET** server control onto a Web form.
2. Drag three Literals onto the Web form.




3. Open the code behind for the Web form.
4. Add the following code to the Page_Load event.

```
Literal1.Text = ContentBlock1.IsLoggedIn

  If ContentBlock1.IsLoggedIn Then

   Literal2.Text = ContentBlock1.loggedInUserName & " is logged in "

   Literal3.Text = ContentBlock1.loggedInUserID & " is the User ID "

  End If
```

5. Build and run the solution.
6. Browse to the login page and log in.
7. Browse to the new Web form you added.
8. The login information is displayed.




### *Accessing Additional Properties*

In addition to the standard properties, **Ektron CMS400.NET** provides access to additional properties for the following objects.

- ListSummary
- Collection
- Search
- ContentBlock
- FormBlock

To access additional properties, use the same syntax you use for standard properties but add `.ekitem` or `.ekitems` after the object. Here is an example.

```
dim MyCB as New ContentBlock

label1.text = MyCB.EkItem.dateCreated
```

or

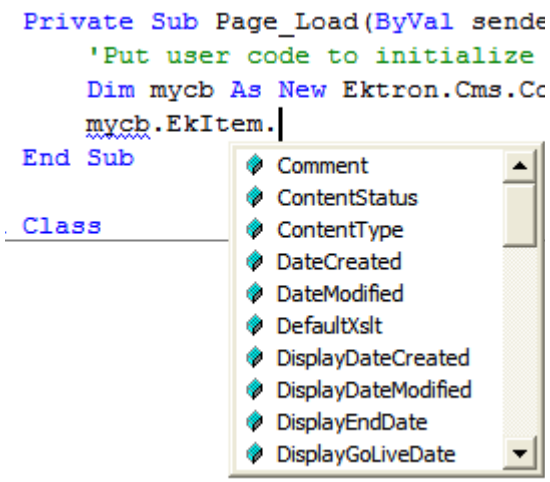
```
dim MyCB as new Ektron.Cms.Controls.ContentBlock

label1.text = MyCB.EkItem.dateCreated
```

**IMPORTANT!** To access additional properties for the Collection, ListSummary, and Search objects, use `ekitems`, not `ekitem`. For example: `MyColl.ekitems(0).dateCreated`. where (0) is the index of the array. For more Information on using ekitems, see "Accessing Items in an Array" on page 47.

With C#, use this syntax.

```
Ektron.Cms.Controls.ContentBlock MyCB = new

Ektron.Cms.Controls.ContentBlock();

label1.text = MyCB.EkItem.dateCreated
```

You can use intellisense to select from a list of additional object properties, as shown below.I




The additional properties are listed below.

**NOTE** The following properties are read-only. For example, you can get a content block's ID and pass it through to another part of the code, however you cannot set a content blocks ID to be shown. `mycb.Ekitem.id = 8` will not set a content block's ID. The correct way to set a content blocks ID is `DefaultContentID = 8`.

| Property | Description | For more information, see |
|---|---|---|
| Comment | The content block's comment | |
| ContentStatus | The status of the content block.<br>• approved<br>• checked out<br>• checked in<br>• expired<br>• pending deletion<br>• pending expiration<br>• pending start date<br>• submitted | **Ektron CMS400.NET** User Manual, "Appendix A: Content Statuses." |
| ContentType | One of the following<br>• all types<br>• archived content<br>• archived forms<br>• content<br>• forms | To learn about archived content, see the **Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting Archive Options." |
| DateCreated | The date when the content block was created, formatted as a .NET date type | |
| DateModified | The date when the content block was modified, formatted as a .NET date type | |
| DefaultXslt | The default Xslt | |
| DisplayDateCreated | The date when the content block was created. It is formatted as a string that represents Ektron CMS400's display of the date. | |

| Property | Description | For more information, see |
|---|---|---|
| DisplayDateModified | The date when the content block was edited. It is formatted as a string that represents Ektron CMS400's display of the date. | |
| DisplayEndDate | The content block's end date. It is formatted as a string that represents Ektron CMS400's display of the date. | **Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting an End Date on a Content Block" |
| DisplayGoLiveDate | The content block's start date. It is formatted as a string that represents Ektron CMS400's display of the date. | **Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting a Start Date" |
| DisplayStartDate | The content block's start date. It is formatted as a string that represents Ektron CMS400's display of the date. | **Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting a Start Date" |
| EndDate | The content block's end date, formatted as a .NET date type | **Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting an End Date on a Content Block" |
| EndDateAction | • archive display<br>• archive expire<br>• refresh report | **Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting Archive Options." |
| FolderID | The ID of the folder that contains each content block | |
| GoLiveDate | The content block's start date formatted as a .NET date type | **Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting a Start Date" |
| Html | The content that makes up the content block. If content block is in XML it will return it as raw XML content | |
| Hyperlink | Content block title wrapped by `<a href>` tags | |

| Property | Description | For more information, see |
|---|---|---|
| Id | The content block 's ID number | |
| InheritedFrom | If folder permissions are inherited, the folder from which they are inherited | **Ektron CMS400.NET** Administrator manual "Setting Permissions" > "Inheritance" |
| IsInherited | Whether a content block's permissions are inherited | **Ektron CMS400.NET** Administrator manual "Setting Permissions" > "Inheritance" |
| IsPrivate | Whether or not a content block is private | **Ektron CMS400.NET** Administrator manual "Managing Users & User Groups" > "Private Content & Memberships" |
| Language | The content block's language | **Ektron CMS400.NET** Administrator manual "Multi-Language Support" > "Working with Multi-Language Content" |
| LastEditorFname | The first name of the last person to edit the content block | |
| LastEditorLname | The last name of the last person to edit the content block | |
| PackageDisplayXSLT | If the content block is XML, the name of its XSLT | |
| QuickLink | The content block's quicklink | **Ektron CMS400.NET** User manual "Library Folder" > "Quicklinks" |
| StartDate | The content block's start date formatted as a .NET date type | **Ektron CMS400.NET** User Manual, "Scheduling Content to Begin and End" > "Setting a Start Date" |

| Property | Description | For more information, see |
|---|---|---|
| Status | The status of the content block<br>• approved<br>• checked out<br>• checked in<br>• expired<br>• pending deletion<br>• pending expiration<br>• pending start date<br>• submitted | **Ektron CMS400.NET** User manual “Appendix A: Content Statuses” |
| Teaser | The content block summary | “Adding a Content Block Summary” in the **Ektron CMS400.NET** User manual |
| TemplateLink | Currently empty and not being used with the ContentBlock server control | |
| Title | The content block title | |
| UserID | Last user who edited the content | |
| Xslt1 | The content block’s first Xslt, as defined in **Ektron CMS400.NET** | |
| Xslt2 | The content block’s second Xslt, as defined in **Ektron CMS400.NET** | |
| Xslt3 | The content block’s third Xslt, as defined in **Ektron CMS400.NET** | |
| Xslt4 | The developer can use this property programmatically. **Ektron CMS400.NET** only uses Xslt1, 2 and 3 in the workarea. | |
| Xslt5 | The developer can use this property programmatically. **Ektron CMS400.NET** only uses Xslt1, 2 and 3 in the workarea. | |

### *Accessing Items in an Array*

To access and manipulate content blocks returned by an object, use the common class `Ektron.Cms.Common.ContentBase`. *EkItems* is an array of Ektron.Cms.Common.ContentBase. *EkItem* is a single Ektron.Cms.Common.ContentBase.

Search, Collection and ListSummary have EkItems (an array of ContentBase), while and ContentBlock have a single EkItem. Here is an example of how to use this feature.

```
dim MyC as new Ektron.Cms.Controls.Collection

MyC.DefaultCollectionID = 1

MyC.Fill()

dim item as Ektron.Cms.Common.ContentBase

MyC.Text = "<ul>"

for each item in MyC.EkItems

    MyC.Text &= "<li>" & item.Title & "</lI>"

next

MyC.Text &= "</ul>"

Response.Write(MyC.Text())
```

This example formats every item in the collection in a bulleted list.

### *Referencing the Page Property*

If a control requires a reference to its parent page (for example, utilizing the DynamicParameter property on a content block to check for a query string), you must provide access to the page object if you declared your control in the code behind. To do this, set the control's page property to the Web page you're working on. For an example, see the code in red and italics below.

```
dim search as new Ektron.Cms.Controls.Search()

MySearch.Page = me

MySearch.Fill()
```

With C#, use this syntax.

```
Ektron.Cms.Controls.Search MySearch = new
Ektron.Cms.Controls.Search();

MySearch.Page = this;

MySearch.Fill();
```

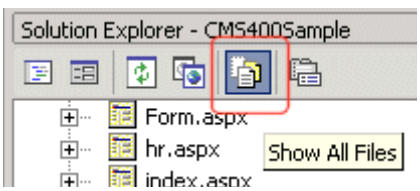
This relationship is only required when inserting a control in the code behind. When dragging and dropping, even if you make changes in code behind, the relationship is automatically generated.

> Best Practice
>
> Ektron recommends including the page property reference when using the server controls as components in the code behind.

## Customizing Server Controls

One of Visual Studio .NET's strengths is its separation of coding and logic from presentation. Web page formatting is handled by a page's HTML, while the logic is handled by the code behind, which is stored in the corresponding .vb file. For example, if the ASP.NET page is mypage.aspx, the code behind file is mypage.aspx.vb.

**NOTE** If you do not see the code behind files, click the Show All Files button on the VS .NET Solution Explorer toolbar.




Within the vb file, you can use Visual Basic to insert code to manipulate the events that occur on the page.

This subsection contains the following topics:


- "Recognizing the Server Control Within the HTML" on page 49
- "Recognizing the Server Control Within the Code Behind" on page 49
- "Customizing the Server Control in the Code Behind" on page 50

- "Troubleshooting Error Creating Control Message" on page 50


### *Recognizing the Server Control Within the HTML*

Within a Web page's HTML, a <cms> tag wraps the VS. NET object, as shown below.

```
<cms:Search id="Search1" runat="server" ButtonText="Search"
Display="Vertical"></cms:Search>
```

Above is an example of a Search Server Control. Below is an example of a ContentBlock Server Control.

```
<cms:ContentBlock id="ctrlMainContentBlock" runat="server"
DefaultContentID="1" DynamicParameter="id" OverrideXslt="Default"></
cms:ContentBlock>
```

### *Recognizing the Server Control Within the Code Behind*

Within the VS .NET code behind file, the Ektron server controls appear (along with the VS .NET controls) in the Web Form Designer Generated Code section. When you click the plus sign (+) to display this section, you see something like the following. The content block listed in HTML above is circled below to help you see their relationship.

```
#Region " Web Form Designer Generated Code "

    'This call is required by the Web Form Designer.
    <System.Diagnostics.DebuggerStepThrough()> Private Sub InitializeComponent()...
    Protected WithEvents MetaDataArea As System.Web.UI.WebControls.Literal
    Protected WithEvents DhtmlJavaScript As System.Web.UI.WebControls.Literal
    Protected WithEvents DropDownMenu As System.Web.UI.WebControls.Literal
    Protected WithEvents ctrlMainContentBlock As Ektron.Cms.Controls.ContentBlock
    Protected WithEvents ctrlTopContentBlock As Ektron.Cms.Controls.ContentBlock
    Protected WithEvents ctrlBottomLeftContentBlock As Ektron.Cms.Controls.ContentBl
    Protected WithEvents ctrlBottomRightContentBlock As Ektron.Cms.Controls.ContentB
    Protected WithEvents ctrlLanguageSelect As Ektron.Cms.Controls.LanguageSelect
    'NOTE: The following placeholder declaration is required by the Web Form Designe
    Private designerPlaceholderDeclaration As System.Object
```

The next section of the code behind page loads the page into the browser.

```
Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
```

We want our events to occur while the page is loading, so we'll add custom code following this line.

### *Customizing the Server Control in the Code Behind*

To customize an Ektron server control in the code behind, insert code similar to the following *after* the Page_load command.

```
Dim MyObj As New Ektron.Cms.Controls.ContentBlock
```

This code declares a variable named MyObj and assigns to it the value of a content block. The content block is part of the Ektron.CMS.Controls content base, so it has access to the **Ektron CMS400.NET** database.

After defining MyObj as a content block, you can access its properties. For example, to assign a defaultID of 24, insert the following.

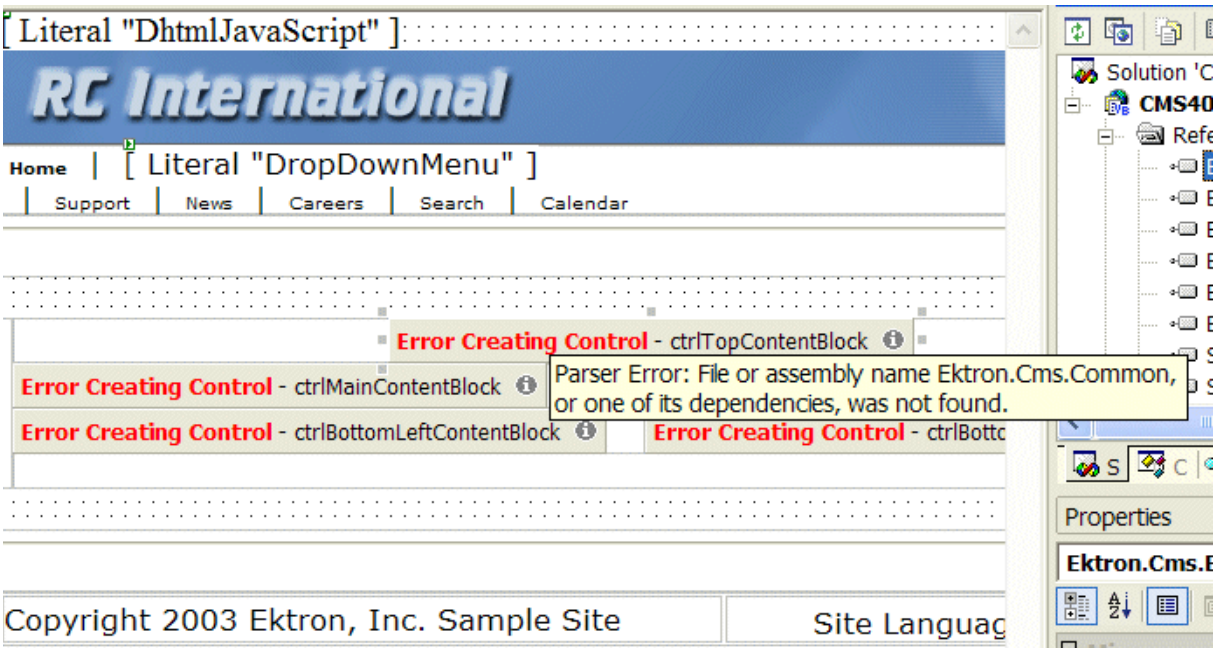
```
Dim MyObj As New Ektron.Cms.Controls.ContentBlock

MyObj.DefaultContentID = 24
```

Now, the content block can be specified dynamically in the URL of the hyperlink that calls it. If not, content block 24 displays.

This is just an example of programmatically applying property values to content blocks. For a complete list of properties available to **Ektron CMS400.NET** objects, see "List of Server Controls" on page 61.

### *Troubleshooting Error Creating Control Message*

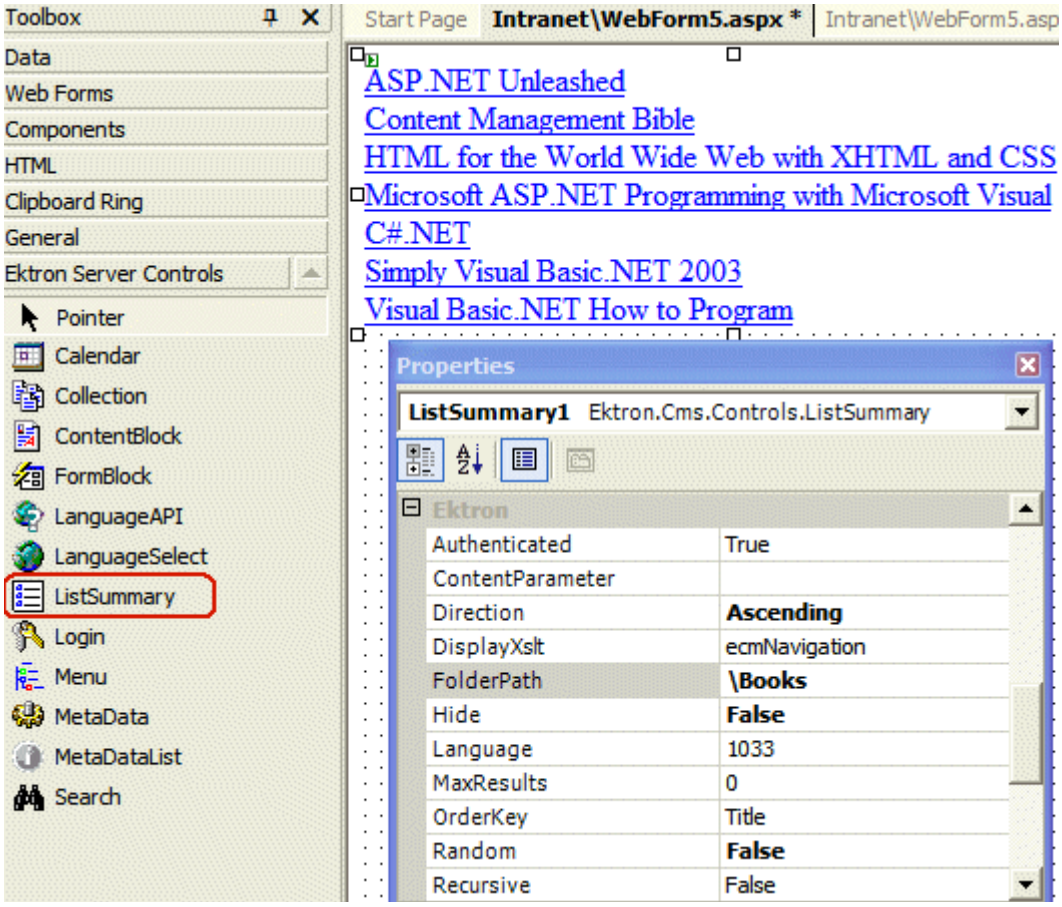
If you get an Error Creating Control message while trying to use a server control, you can view the text of the error message by hovering the mouse over the control. See illustration below.


[ Literal "DhtmlJavaScript" ]
RC International
Home | [ Literal "DropDownMenu" ]
Support | News | Careers | Search | Calendar
Error Creating Control - ctrlTopContentBlock
Error Creating Control - ctrlMainContentBlock
Parser Error: File or assembly name Ektron.Cms.Common, or one of its dependencies, was not found.
Error Creating Control - ctrlBottomLeftContentBlock
Error Creating Control - ctrlBottc
Copyright 2003 Ektron, Inc. Sample Site
Site Languag
Solution 'C
CMS40
Properties
Ektron.Cms.E

# Using Drag & Drop and Programmatically Together

All server controls can be further manipulated after dragging and dropping them on a web form by using the code behind. Here is an example of using a drag and drop ListSummary Server Control then modifying it programmatically.

First, drag and drop a List Summary on your Web form and set your properties.




Then add the following code to the code behind.

```
Dim myString As String
Dim i
For i = LBound(ListSummary1.EkItems) To UBound(ListSummary1.EkItems)
myString &= "<a href=""" & ListSummary1.EkItems(i).QuickLink & """>" &
ListSummary1.EkItems(i).DateCreated & "</a><br>"
            ListSummary1.Text = myString
Next
```

An Explanation of the code:

ListSummary1 is the ID of the object. It is used to get access to its properties.

Create a string that contains the output (myString).

```
Dim myString As String
```

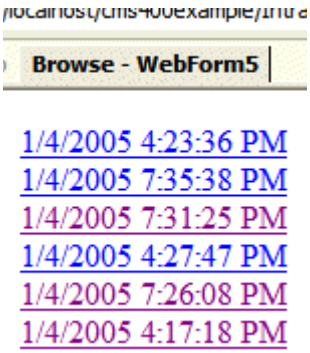
Set the object's text property to that string.

```
myString &= "<a href=""" & ListSummary1.EkItems(i).QuickLink & """>"
& ListSummary1.EkItems(i).DateCreated & "</a><br>"
```

Wrap in a loop so it loops through each of the items.

```
Dim i

For i = LBound(ListSummary1.EkItems) To UBound(ListSummary1.EkItems)

Next
```

This example outputs the date created for each content block in a List Summary.




# Data Binding with Server Controls

With data binding, you can bind Ektron server controls to a DataGrid Control, DataList Control, or Repeater Control. This gives more flexibility when you use data from the Ektron server controls. Some benefits of data binding are ease of data manipulation and the ability to format data.

| Title | ID | Teaser |
|---|---|---|
| Plastic Molder #123 | 13 | RC International is looking for an experienced plastics molder. |
| Servo Control Engineer #124 | 14 | RC International is looked for a highly skilled servo control eng |

The following are data bindable Ektron Server Controls.

- Collection
- Content Block
- Form Block
- List Summary
- Search

The following is an example of code behind that uses a DataGrid to display a Collection:

```
Dim myCol As New Ektron.Cms.Controls.Collection

myCol.DefaultCollectionID = 1

myCol.Fill()

DataGrid1.DataSource = myCol

DataGrid1.DataBind()
```

**NOTE** When using drag and drop server controls for the Collection and DataGrid, you need to call the DataBind() method in the code behind. An example line of code would be: `DataGrid1.DataBind()` .

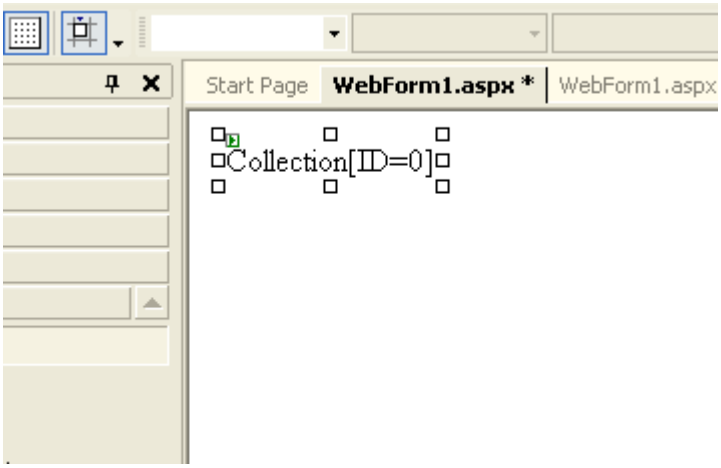
This subsection contains the following topics:


- ”Steps to Data Binding Using Drag and Drop Server Controls” on page 54
- ”Customizing the DataGrid Table View” on page 57

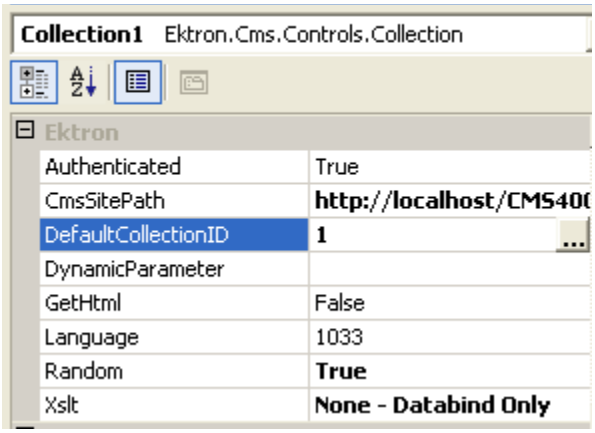

## Steps to Data Binding Using Drag and Drop Server Controls

The following example uses a Collection.

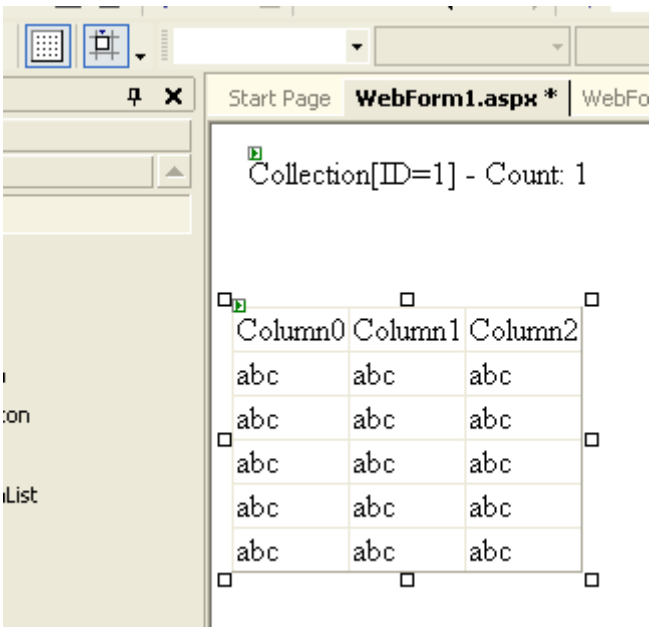
1. Drag and drop a data bindable Server Control on the Web form.



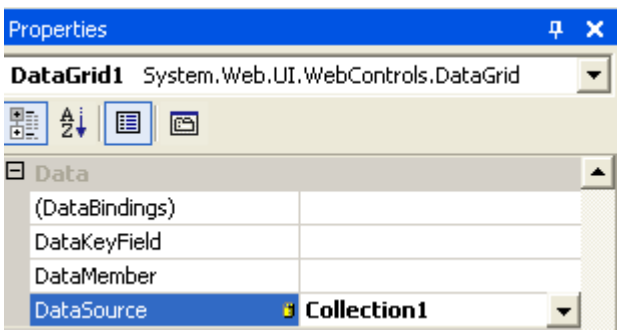

2. In properties, choose the Ektron default ID for the Server Control and make sure the XSLT is set to **None- DataBind Only**.



3. Drag and drop a DataGrid on the web form.




4. In the DataGrid properties, choose the DataSource.




5. The DataGrid is populated with column headers and the type of information that will be in the table.
6. Go to code behind and add `DataGrid1.DataBind()` to the Page_Load event.

```
Private Sub Page_Load(ByVal sender As System.Object
    'Put user code to initialize the page here
    DataGrid1.DataBind()
End Sub
```

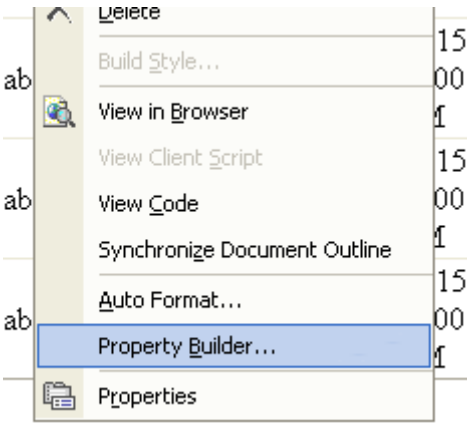
7. Switch to Design mode and right-click on the DataGrid. Then, choose **View in Browser**. The table appears.

| LastEditorFname | DateModified | Html | QuickLink | HasErrors | Teaser | ID |
|---|---|---|---|---|---|---|
| Application | 2/17/2004 4:12:11 PM | | /CMS400Sample/products.aspx?id=5 | False | RC Internationals most successful entry in to the competitive RC racing world is the RC Cheetah. | 5 |

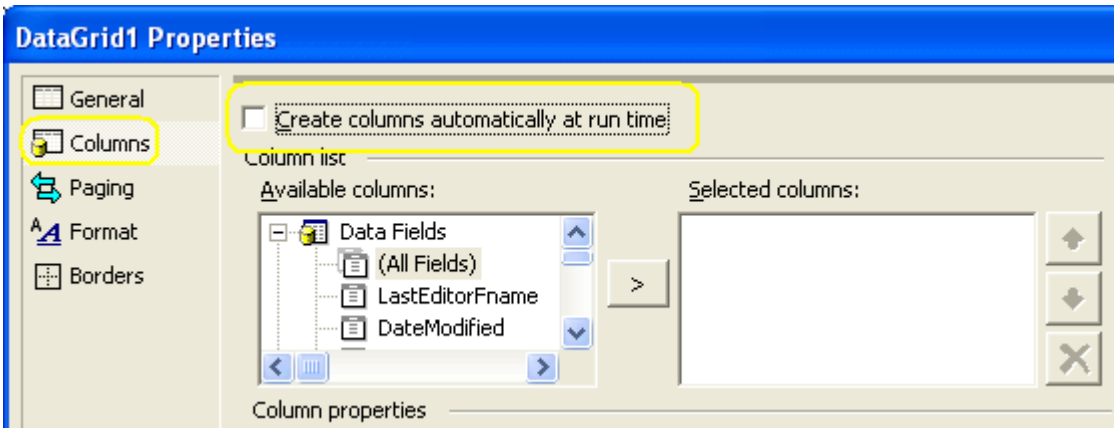
## Customizing the DataGrid Table View

Customizing the DataGrid is a great way to display and format specific information from a data source. You can select formatting for columns, paging and borders, plus other general formatting for the table. In the following example, we choose table columns.

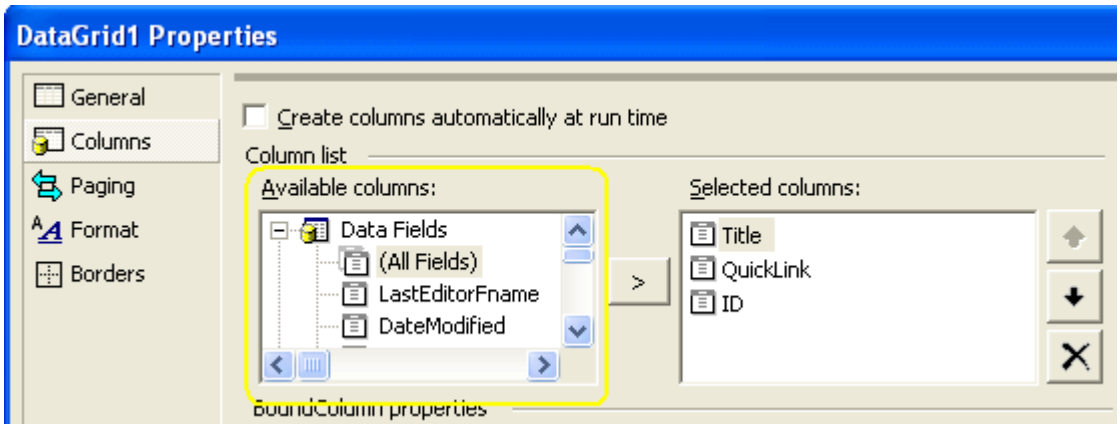
1. In Web form design, right click the DataGrid and choose **Property Builder**.




2. Click **Columns** and uncheck the **Create columns automatically at run time** checkbox.



3. Choose the fields for the custom table from the available columns.




4. Click **OK**.
5. Right click the DataGrid and choose **View in Browser**.
6. The custom table appears.

| Title | QuickLink | ID |
|---|---|---|
| RC Lilly | /CMS400Sample/products.aspx?id=6 | 6 |

For more information on working with DataBind() or DataGrid, review the help topics in Visual Studio.

# Caching with Server Controls

One of the most important factors in building high-performance, scalable Web applications is the ability to store items in memory the first time they are requested. These items include data objects, pages, or parts of a page.

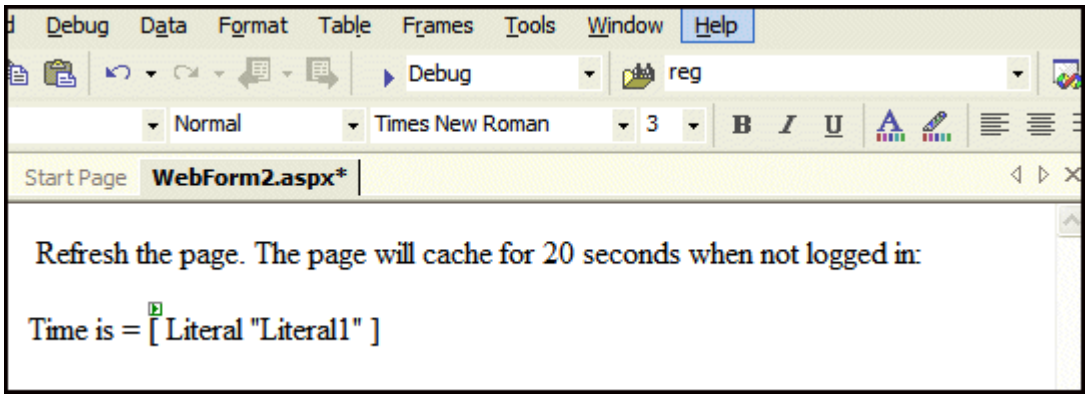
Known as caching, this process allows you to use a number of techniques to store page output or application data across HTTP requests and reuse it. You can store items on the Web server or other software in the request stream, such as the proxy server or browser.

With caching, the server does not have to recreate information, particularly information that demands significant processor time or other resources.This saves valuable time and resources.

Below is an example of a page level cache for non logged in users. In this example, you use a Server Control to define whether a user is logged in or not. Then, further define if the time is cached, based on the user's status.

1. Create a new Web form in your Ektron CMS400.NET project.
2. Add the following text to the Web form:
   **Refresh the page. The page will cache for 20 seconds when not logged in:**

   **Time is =**
3. Next to **Time is =**, add a literal.




4. Below that, add a ContentBlock Server Control.
5. Set the DefaultContentID. For this example, DefaultContentID = 20.
6. Add the following code to the code behind.

```
If Not ContentBlock1.IsLoggedIn Then

            Response.Cache.SetExpires(DateTime.Now.AddSeconds(20))

            Response.Cache.SetCacheability(HttpCacheability.Public)

            Response.Cache.SetValidUntilExpires(True)

            Response.Cache.VaryByParams("id") = True

            Response.Cache.SetVaryByCustom("cmsCache")

        End If

        Literal1.Text = Now()
```

7. Build your Web form.
8. Browse to your Web form with your browser.

9. Press the refresh button on your browser.

If you are not logged in to CMS400.NET, the time remains the same for twenty seconds. After twenty seconds, when you refresh, you get the new time.

# List of Server Controls

To customize your Ektron CMS400.NET application, you can use these server controls.

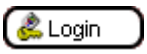
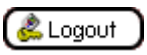
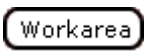
| Server Control Names | Description | More Information |
|---|---|---|
| BreadCrumb | Creates a breadcrumb trail of where site visitors have been and allows them to navigate back to previous pages. | "BreadCrumb Server Control" on page 198 |
| Calendar | Displays an event calendar. | "Calendar Server Control" on page 127 |
| Collection | Displays links generated by collections. | "Collection Server Control" on page 120 |
| ContentBlock | Displays and manages a content block. | "ContentBlock Server Control" on page 71 |
| DhtmlMenu | Displays a drop down menu with submenus | "DhtmlMenu Server Control" on page 178 |
| FormBlock | Displays a content block that is associated with a form. | "FormBlock Server Control" on page 81 |
| IndexSearch | Perform XML index searches on XML content. | "IndexSearch Server Control" on page 97<br><br>For information on using XHTML Search, see "Search Server Control" on page 85 |
| LanguageAPI | Allows developers to force a language for a Web site. | "LanguageAPI Server Control" on page 68 |
| LanguageSelect | Lets user select language of site. | "LanguageSelect Server Control" on page 65 |

| Server Control Names | Description | More Information |
|---|---|---|
| ListSummary | Displays a list of content block summaries. | ”ListSummary Server Control” on page 114 |
| Login | Displays the login or logout button, which lets the user log in and use Ektron CMS400.NET. | ”Login Server Control” on page 63 |
| Menu | Calls a defined menu to be used with an Xslt to display the menu. | ”Menu Server Control” on page 161 |
| MetaData | Adds metadata contained in content blocks to a Web page. | ”MetaData Server Control” on page 186 |
| MetaDataList | Creates a list of content blocks based on the metadata contained in the content blocks. | ”MetaDataList Server Control” on page 189 |
| RssAggregator | Aggregates an RSS feed from a RSS Web Site. | ”RssAggregator Server Control” on page 194 |
| Search | Inserts a text box with a search button in the template. The site visitor enters criteria and launches a search of the XHTML content on your Web site. | ”Search Server Control” on page 85<br><br>For information on using XML Indexing search, see ”IndexSearch Server Control” on page 97 |

# Login Server Control

The Login Server Control paints a login button on the template when displayed in a browser.

When the Login Server Control is inserted and the project is built, the control displays the following buttons on a Web page.

| Button | Description |
|---|---|
|   | When user is not logged in, this button appears. Clicking the button opens the login window, where a user can enter a username and password. Upon authentication, the user is logged in to the Ektron CMS400 Web site. |
|   | After a user logs in, this button replaces the login button to let the user log out. |
|   | When logged in, this button appears under the logout button, allowing the user to access the Workarea. |
| | Lets the user preview the entire Web site as if all checked-in content were published. For more information, see the **Ektron CMS400.NET** User Manual section “Logging In and Out” > “Site Preview.” |
| | Turns off site preview mode. |

This subsection contains the following topics:


- ”Placing a Login Button” on page 63
- ”Login Server Control Properties” on page 64


## Placing a Login Button

You can add any number of login buttons to a template. You can insert a login button on each template, or set up a special Web

page, called login.aspx, from which users can log into the CMS400 site without the public being able to access the page.

# Login Server Control Properties

The following table explains the properties of the Login Server Control.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | String |
| Hide | Used to hide login button in design time and run time.<br>**True** = Hide login button<br>**False** = Show login button | Boolean |
| Language | Design time only: View content blocks, collections, etc. in different languages<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |

# LanguageSelect Server Control

The LanguageSelect Server Control displays a language selection dropdown list on an Ektron CMS400.NET Web page.

This control lets a visitor to your Web site select a language in which to view the site. Here is what the control looks like when published on a Web page.

It lists all languages selected on the **Settings > Configuration > Language settings** screen. (For more information, see the "Multi-Language Support "chapter of the **Ektron CMS400.NET** Administrator manual.)

You can place this control in any location of any page on your site. On the sample site, it appears in the lower right corner of the home page.

This subsection contains the following topics:


- "LanguageSelect Server Control vs. LanguageAPI Server Control" on page 65
- "LanguageSelect Server Control Properties" on page 66


## LanguageSelect Server Control vs. LanguageAPI Server Control

The LanguageSelect Server Control lets users view your site in the language they choose. The LanguageAPI Server Control, on the other hand, forces users to view a site in a particular language. For

information on LanguageAPI, see ”LanguageAPI Server Control” on page 68.

# LanguageSelect Server Control Properties

The following table explains the properties of the LanguageSelect control.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* ”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 32 | String |
| Hide | Used to hide language select dropdown box and label in design time and run time.<br>**True** = Hide language select dropdown box and label<br>**False** = Show language select dropdown box and label | Boolean |
| LabelName | Lets you define the label next to the language select dropdown box.<br>The default label is **Language** | String |
| Language | Design time only: View content blocks, collections, etc. in different languages<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |

| Property | Value | Data Type |
|---|---|---|
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |

# LanguageAPI Server Control

The LanguageAPI Server Control lets a developer force a particular language for a Web site. You can do this simply by dropping the server control on the page and then choosing a language in the SiteLanguage property box. You can also override site language logic by programmatically using the LanguageAPI Server Control to detect the browsers language, and display the site in that language.

This subsection contains the following topics:


- ”LanguageSelect Server Control vs. LanguageAPI Server Control” on page 68
- ”LanguageAPI Server Control Properties” on page 68
- ”Code Behind Only Properties” on page 69


## LanguageSelect Server Control vs. LanguageAPI Server Control

The LanguageSelect Server Control lets users view your site in the language they choose. The LanguageAPI Server Control, on the other hand, forces users to view a site in a particular language. For information on LanguageSelect, see ”LanguageSelect Server Control” on page 65.

## LanguageAPI Server Control Properties

The LanguageAPI Server Control properties are described in this table.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| SiteLanguage | Sets the site language. Runs through the IsValid function to verify that the language is an active language in the system. | String |
| SiteLanguageID | Numeric value of the site language. This is the property you will use if you are using code behind to set the sites language ID.<br>Example: 1036 = French<br>For a list of supported languages, see The Administrator Manual > “Appendix A: List of Supported Languages” | Integer |

# Code Behind Only Properties

The following is a list of LanguageAPI specific properties that can only be used programmatically.

| Property | Value | Data Type |
|---|---|---|
| DefaultLanguage | **Read only**. What the default language of the site is set to. For example, the demo site is “English (Standard)”. | String |
| DefaultLanguageID | **Read only**. Returns the value that is the default language id of the site. For example, the demo site is “1033” for English. | Integer |
| GetLanguage | This method returns a string. Pass in a valid language ID and it will return the language name. | Integer Argument |
| GetLanguageID | This method returns an integer. Pass in a valid language name and it will return the language ID. | String Argument |
| IsValid | This method returns a boolean. You can pass in a language ID or a string and it will tell you if the system is supporting it. | Argument |

| Property | Value | Data Type |
|---|---|---|
| LanguageIdList | **Read only**. Lists all the language IDs that are activated in **Ektron CMS400.NET**.<br>For information on how to enable languages, see The Administrator Manual > “Multi-Language Support” > “Working with Multi-Language Content” > “Determining Which Languages are Available”. | Array of Integers |
| LanguageTitleList | **Read only**. Lists all the languages that are activated in **Ektron CMS400.NET**. | Array of Strings |
| MultiLanguageEnabled | **Read only**. Tells if the site supports multi-language mode.<br>**True** = Multi-language enabled<br>**False** = Multi-language not enabled<br>Multi-language is enable by default in **Ektron CMS400.NET**. To disable Multi-language, see Administrator Manual > “Multi-Language Support” > “Working with Multi-Language Content” > “Disabling Support for Multiple Language Content”. | Boolean |

# ContentBlock Server Control

The ContentBlock Server Control displays a content block on an Ektron CMS400.NET Web page.

Ektron CMS400.NET has two types of content blocks:

- Static - displays one specified content block
- Dynamic - displays the content block of the ID passed through a URL parameter

In addition, a content block can contain XHTML or XML content.

For an example of using the ContentBlock Server Control programmatically, see "Using the ContentBlock Server Control Programmatically Example" on page 79

This subsection contains the following topics:


- "Static Content Block" on page 71
- "ContentBlock Server Control Properties" on page 72
- "Dynamic Content Block" on page 73
- "XML Content Block" on page 76
- "The OverrideXslt and DisplayXslt Properties" on page 77
- "Using the ContentBlock Server Control Programmatically Example" on page 79


## Static Content Block

A static content block displays one identified content block on a Web page. The following table explains how to complete the server control properties to insert a static content block on a Web page.

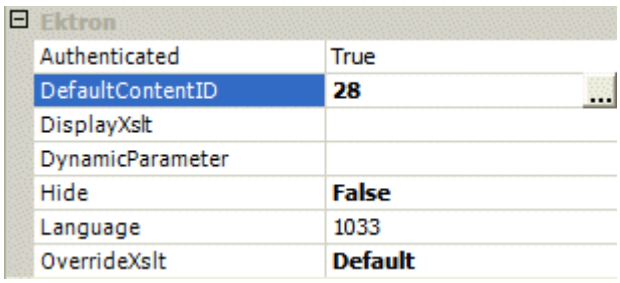
## ContentBlock Server Control Properties

The following table explains the properties of the ContentBlock Server Control.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | String |
| DefaultContentID | The ID of a content block that appears where you insert this server control.<br>If you don't know the ID number of the content block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | Integer |
| DynamicParameter | Select **None - Use Default**. This parameter is used for dynamic content blocks. | String |
| Hide | Used to hide a content block in design time and run time.<br>**True** = Hide content block<br>**False** = Show content block | Boolean |
| Language | Design time only: View content blocks, collections, etc. in different languages<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |

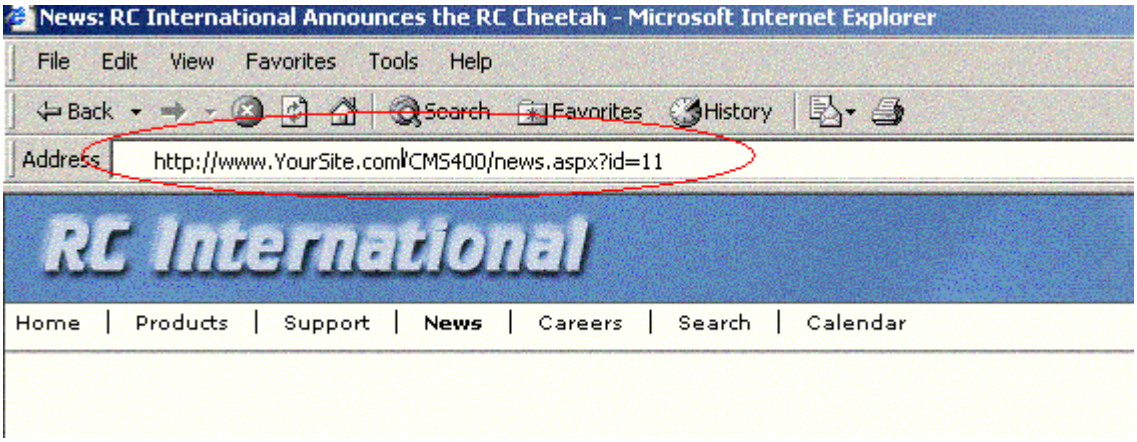
| Property | Value | Data Type |
|---|---|---|
| OverrideXslt | Ignore for a non-XML content block. | must be one of these values: default, one, two, three |
| DisplayXslt | Ignore for a non-XML content block. | String |
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |




In the above example, the server control retrieves content block id=28 and displays it in the browser.

# Dynamic Content Block

Use a dynamic content block to display a content block whose ID is passed through a URL parameter. You would use this server control with a dynamic template.


News: RC International Announces the RC Cheetah - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://www.YourSite.com/CMS400/news.aspx?id=11
RC International
Home | Products | Support | News | Careers | Search | Calendar

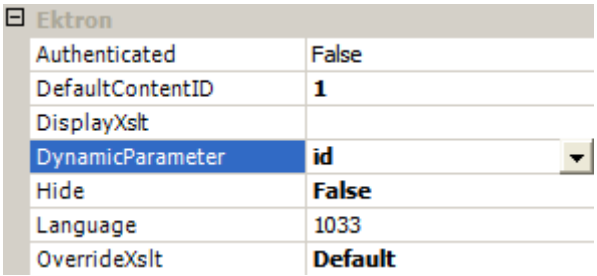
The following table explains how to complete the ContentBlock Server Control's properties to insert a dynamic content block on a template.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | String |
| DefaultContentID | The ID of a content block that appears where you inserted this server control if no other content block is identified, or is not available.<br>If you don't know the ID number of the content block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | Integer |
| DynamicParameter | Select **id**. When you do, this server control uses the content block passed as a URL parameter. | String |

| Property | Value | Data Type |
|---|---|---|
| Hide | Used to hide a content block in design time and run time.<br>**True** = Hide content block<br>**False** = Show content block | Boolean |
| Language | Design time only: View content blocks, collections, etc. in different languages<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |
| OverrideXslt | Ignore for a non-XML content block. | must be one of these values: default, one, two, three |
| DisplayXslt | Ignore for a non-XML content block. | String |
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |

The following example shows how the ContentBlock Server Control can be used to pass the id as a URL parameter.




When a user clicks a link that passes the content block ID as a URL parameter, that content block appears. If that content block is not available, content block 1 appears.

# XML Content Block

Use an XML content block to display an XML content block on an Ektron CMS400.NET Web page.

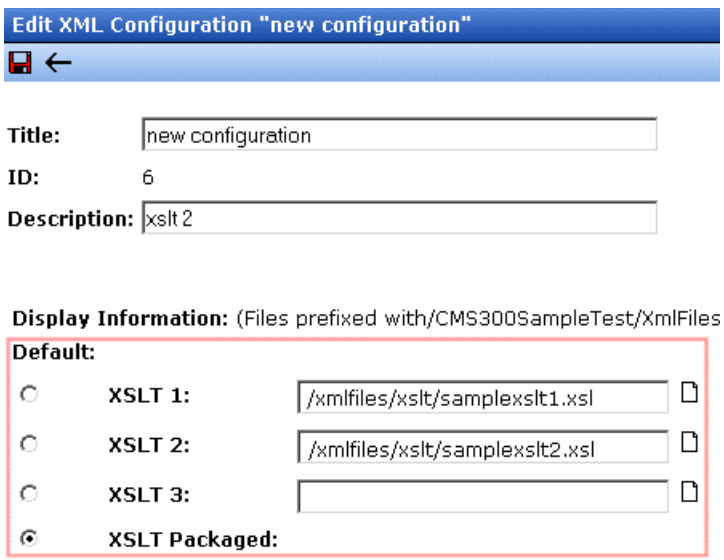
**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | String |
| DefaultContentID | The ID of a content block that appears where you inserted this server control if no other content block is identified, or is not available.<br>If you don't know the ID number of the content block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | Integer |
| DynamicParameter | To make this content block dynamic, select **id**. When you do, this server control uses the content block passed as a URL parameter. | String |

| Property | Value | Data Type |
|---|---|---|
| Hide | Used to hide a content block in design time and run time.<br>**True** = Hide content block<br>**False** = Show content block | Boolean |
| Language | Design time only: View content blocks, collections, etc. in different languages<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |
| OverrideXslt | An internal Xslt file used to display XML content if one is not identified in the Xslt property.<br>*See Also:* "The OverrideXslt and DisplayXslt Properties" on page 77 | must be one of these values: default, one, two, three |
| DisplayXslt | Specify an external XSLT file. *See Also:* "The OverrideXslt and DisplayXslt Properties" on page 77 | String |
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |

## The OverrideXslt and DisplayXslt Properties

The DisplayXslt property is optional. If used, it specifies an external XSLT file. If the DisplayXslt property is not defined, the OverrideXslt property specifies an XSLT identified in the Edit XML Configuration screen. The following table provides more information about these properties.

| To use this display XSLT | Enter this value for XSLT | Enter this value for Override XSLT |
|---|---|---|
| The default XSLT specified in the Edit XML Configuration screen (illustrated below). Note that, in this example, the XSLT Packaged option is the default XSLT, since it is selected.<br>XSLT Packaged is the XSLT from the Edit XML configuration screen (that is, the XSLT created in the Data Designer).<br>  | | Default |
| **XSLT 1** from the Edit XML configuration screen | | 1 |
| **XSLT 2** from the Edit XML configuration screen | | 2 |
| **XSLT 3** from the Edit XML configuration screen | | 3 |

| To use this display XSLT | Enter this value for XSLT | Enter this value for Override XSLT |
|---|---|---|
| An absolute or relative path to an XSLT file. | An external XSLT file not specified in the Edit XML configuration screen.<br>For example<br>`sample.xslt` | does not matter - if XSLT value exists, **OverrideXSLT** is ignored |

*See Also:* The “Managing XML” chapter of the **Ektron CMS400.NET** Administrator manual

# Using the ContentBlock Server Control Programmatically Example

The following code will display a content block:

**NOTE** Before adding these lines of code, you need to drag and drop a literal box on your Web form.

```
Dim MyContentBlock As New ContentBlock

MyContentBlock.DefaultContentID = 8

MyContentBlock.Fill()

Literal1.Text = MyContentBlock.EkItem.Html
```

To display a content block with the content block title, do the following:

**NOTE** Before adding these lines of code, you need to drag and drop a two literal boxes on your Web form.

```
Dim MyContentBlock As New ContentBlock

MyContentBlock.DefaultContentID = 8

MyContentBlock.Fill()
```

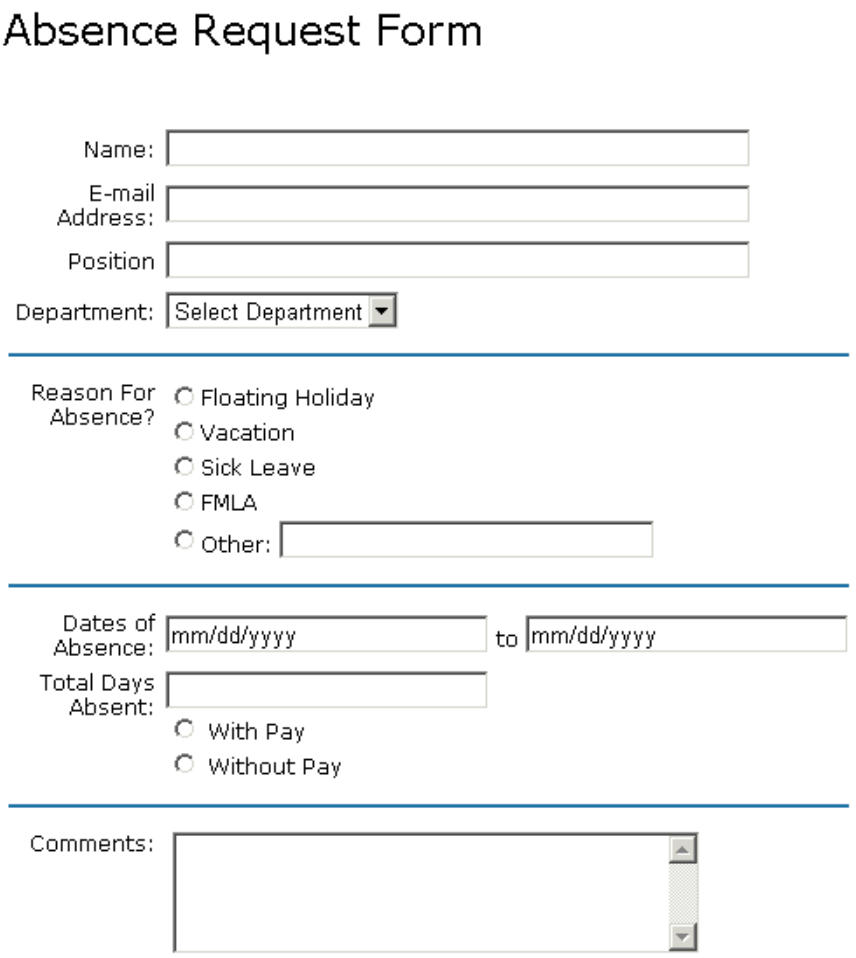
```
Literal1.Text = MyContentBlock.EkItem.Title

Literal2.Text = MyContentBlock.EkItem.Html
```

# FormBlock Server Control

The FormBlock Server Control displays a content block associated with a form.

**IMPORTANT!** If you create a template for an existing form content block, you must manually change its quicklink to point to the new template. This change does not occur automatically. This procedure is described in the Ektron CMS400.NET Administrator manual “Managing Library Assets” > “Updating Default Template for Multiple Quicklinks.”

When added to a template and visited, the form content block might look like this. You can change the display to suit your needs by modifying its properties.




This subsection contains the following topics:


- ”FormBlock Server Control Properties” on page 82
- ”Automatic versus Manual Generation of Form Tags” on page 84


# FormBlock Server Control Properties

The following table explains the properties of the FormBlock Server Control.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| AddValidation | Set to **True** to add Validation to your main runat=server form. | Boolean |
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32. | String |
| DefaultFormID | The ID of a FormBlock that appears where you inserted this server control if no other form block is identified.<br>If you don't know the ID number of the form block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32. | Integer |
| DynamicParameter | To make this form block dynamic, select **id**. When you do, this server control uses the form block passed as a URL parameter. | String |
| FormName | Set this value to the name of your main runat=server form. | String |
| Hide | Used to hide a form block in design time and run time.<br>**True** = Hide form block<br>**False** = Show form block | Boolean |
| Include Tags | Determines if tags are generated automatically of manually. *See Also:* "Automatic versus Manual Generation of Form Tags" on page 84 | Boolean |
| Language | Design time only: View content blocks, collections, etc. in different languages.<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |

| Property | Value | Data Type |
|---|---|---|
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |

# Automatic versus Manual Generation of Form Tags

When using the FormBlock Server control's Include tags property, you have two options.

| Option | Recommended if you are developing | How to Enable |
|---|---|---|
| Let **Ektron CMS400.NET** generate form tags automatically | A pure script .aspx page | Set Include Tags property to **True**. |
| Modify HTML form tags in the .aspx file | An .aspx page and associated code-behind web form | Set Include Tags property to **False**.<br>Here is the default .NET generated form tag:<br>`<form id="Form1" method="post" runat="server">`<br>Modify the form tag as indicated in red:<br>`<form id="Form1" method="post" runat="server" OnSubmit="EkFmValidate(this);">` |

# Search Server Control

The Search Server Control displays a text box with a search button on the template. To learn about which content is considered by this search function, see the **Ektron CMS400.NET** Administrator manual section “Searching Content on the Web Site.”

When added to a template and visited, the Search Server Control looks like this. Users can enter text and click the search button to search for XHTML content on the Web site.

This subsection contains the following topics:


- ”Search Server Control vs. IndexSearch Server Control” on page 86
- ”Search Server Control Properties” on page 86
- ”Search Server Control Property Example” on page 89
- ”Text Box Examples” on page 90
- ”Button Image Source Examples” on page 91
- ”Search Button Text Examples” on page 92
- ”Display Examples” on page 93
- ”ShowDate Examples” on page 93
- ”Specify Search Control To Display Search Results on a Separate Page” on page 94

# Search Server Control vs. IndexSearch Server Control

The Search Server Control lets users search your Web site for XHTML Content. The IndexSearch Server Control lets users search XML content on your Web site. For more information on the IndexSearch Server Control, see "IndexSearch Server Control" on page 97.

# Search Server Control Properties

The following table explains the properties of the Search Server Control.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Description | Data Type | For more information, see |
|---|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to the folder needed for the **FolderPath** property. | String | "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 |
| ButtonImgSrc | If you want to display an image on the submit button, enter the server path to it. | String | "Button Image Source Examples" on page 91 |
| ButtonText | The text used for the button if no image source is identified. If an image source is identified, this is alternative text for the button. | String | "Search Button Text Examples" on page 92 |
| Display | Determines if the search options are in a row (**Horizontal**) or a column (**Vertical**). | either vertical or horizontal | "Display Examples" on page 93 |

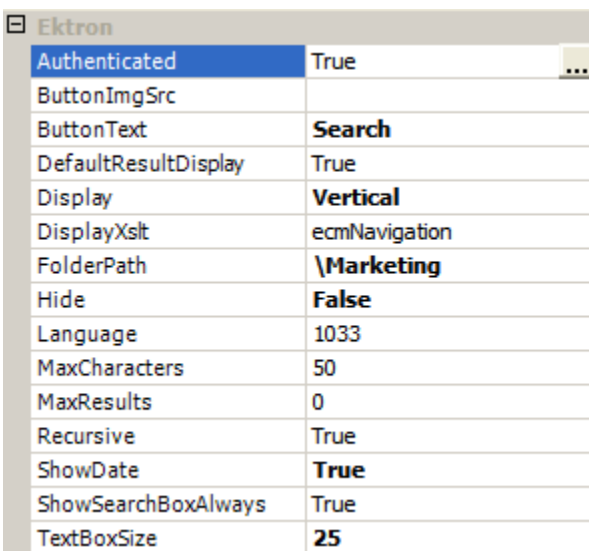
| Property | Description | Data Type | For more information, see |
|---|---|---|---|
| DisplayXslt | Determines how the information on the page is displayed<br>**None**-databind only<br>**ecmNavigation** - lists the title of every content block in the folder<br>**ecmTeaser** - lists the title of every content block in the folder plus the content summary<br>**Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page | String | **ecmNavigation -** "Example of ecmNavigation Display" on page 122<br>**ecmTeaser** - "Example of ecmTeaser Display" on page 123 |
| EmptyResultMsg | The return message displayed, if the search returns no hits.<br>The default is:<br>Search returned zero results | String | |
| FolderPath | The folder at which the search begins. The starting folder need not be the root folder. The Recursive property determines if the search examines this folder's subfolders. | String | "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 |
| Hide | Used to hide Search Server Control in design time and runtime.<br>**True** = Hide control<br>**False** = Show control | Boolean | |
| Language | Design time only: View content blocks, collections, etc. in different languages.<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer | |

| Property | Description | Data Type | For more information, see |
|---|---|---|---|
| MaxCharacters | The maximum number of characters the input text box accepts when doing a search. | Integer | |
| MaxResults | The Maximum number of content blocks returned (0=unlimited). | Integer | |
| OrderDirection | The direction the search results will be sorted in. The default is **Ascending**<br>• **Ascending** - Results from A to Z<br>• **Descending** - Results from Z to A | String | |
| PostResultsAt | Enter the Web form where the search results will be posted. When left blank, the results are posted to the Web form where the search was initiated.<br>**Important:** A search control must exist on the posting page. | String | "Specify Search Control To Display Search Results on a Separate Page" on page 94 |
| Recursive | Whether to search sub-folders of the identified root folder. The root folder is identified in the `Folderpath` property. | Boolean | |
| ShowDate | If **True**, displays the content block's last modified date. This is useful for determining the most recent content block. | Boolean | "ShowDate Examples" on page 93 |
| ShowSearchBoxAlways | If set to **false**, the search box does not appear on PostBack | Boolean | |
| Text Box Size | The size of the display text box for user input. The width is based on the number of characters. | Integer | "Text Box Examples" on page 90 |

| Property | Description | Data Type | For more information, see |
|---|---|---|---|
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String | |

## Search Server Control Property Example

The following example places a text field and a search button on a template.

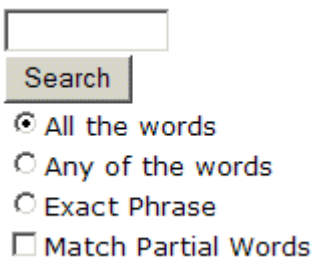
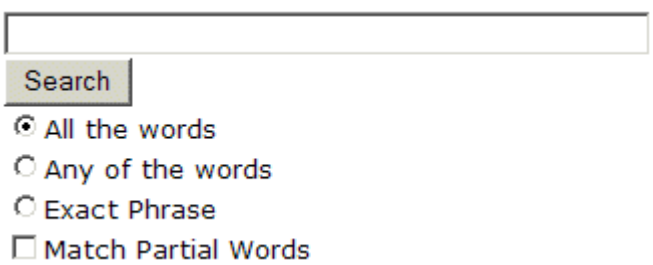



The above example has the following parameters:

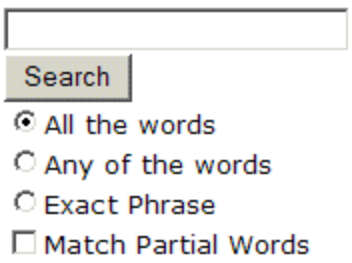
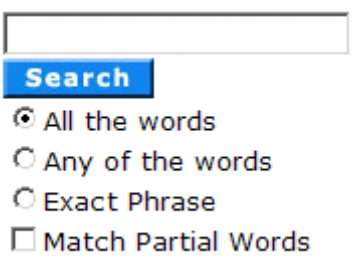
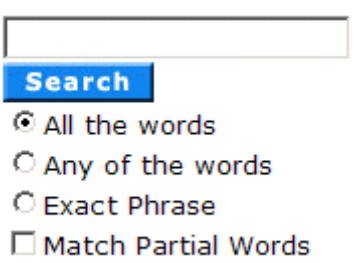
- *No image* is used for the button

- The text on the submit button is ***Search***
- The results will display on the page
- The search options are arranged *vertically*
- Searches content in the ***\content\marketing*** folder
- The search allows a maximum of ***50*** characters
- Returns *unlimited* results when the search is performed
- Search is ***recursive***
- Shows the *last modified date* of the content
- The search box will appear on PostBack
- The size of the text box is ***25*** characters

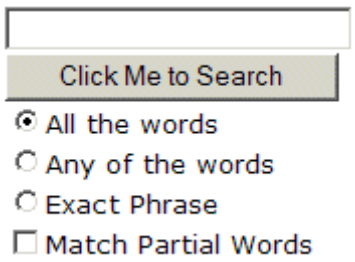
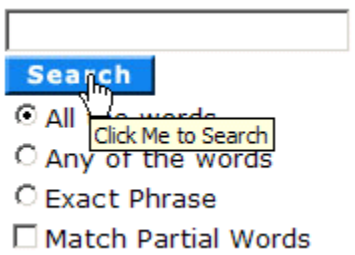
## Text Box Examples

| Value | Web Page View |
|---|---|
| 10 |  |
| 50 |  |

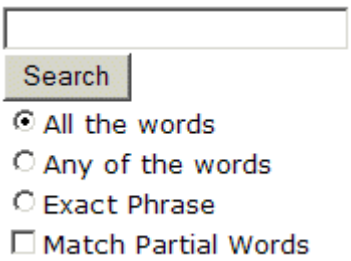
## Button Image Source Examples

| Value | Web Page View |
|---|---|
| (blank) |  |
| search.jpg |  |
| http://www.ektron.com/images/search.jpg |  |

## Search Button Text Examples

| Value | Web Page View |
|---|---|
| “Click Me to Search” |  |
| “Click Me to Search” |  |

## Display Examples

| Value | Web Page View |
|---|---|
| Vertical |  |
| Horizontal | |

## ShowDate Examples

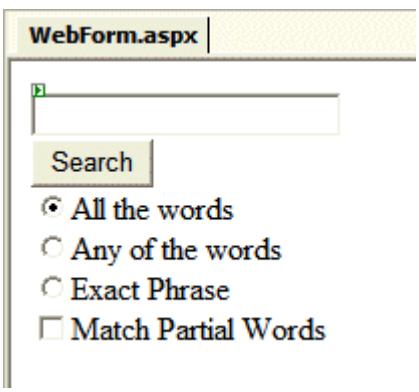
| Value | Web Page View |
|---|---|
| False | |

| Value | Web Page View |
|---|---|
| True | **Search Results**<br><br>RC Sportster *(8/27/2002 7:52:40 PM)*<br>RC Redstar *(8/27/2002 7:52:01 PM)*<br>RC Lilly *(8/27/2002 7:51:26 PM)*<br>RC Cheetah *(8/27/2002 7:50:53 PM)*<br>Products Page *(8/27/2002 7:50:09 PM)*<br>RC International Ships RC Sportster *(8/27/2002 2:46:12 PM)*<br>RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)*<br>RC International Announces the RC Cheetah *(8/27/2002 2:43:01 PM)* |

# Specify Search Control To Display Search Results on a Separate Page

The following example shows a search dialog box on a Web form and the results displayed on separate Web form. This is done when you have the search dialog box in a navigation pane, and the results displayed on a separate Web form.

1. Drag and drop a Search Server Control on a Web form.



2. Set the following Ektron properties for the Search Server Control.

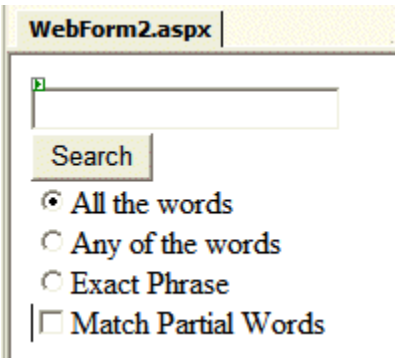
   **DefaultResultsDisplay** = False

   **DisplayXslt** = None - Databind Only

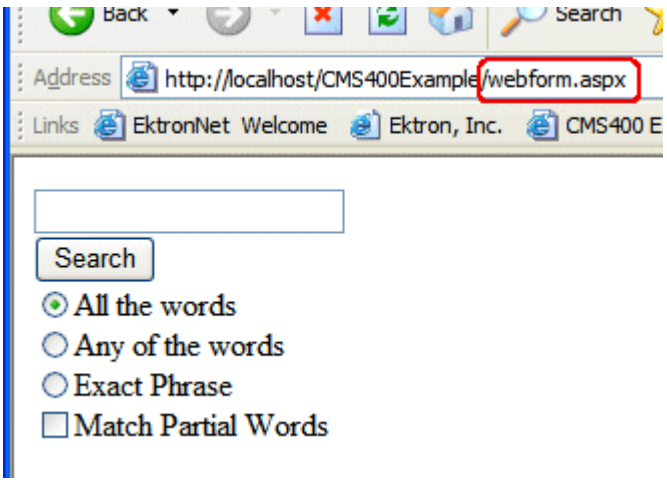
   **PostResultAt** = The Web form name where the results are posted. For example `Webform2.aspx`

**NOTE** webform2.aspx represents the name of the second Web form.

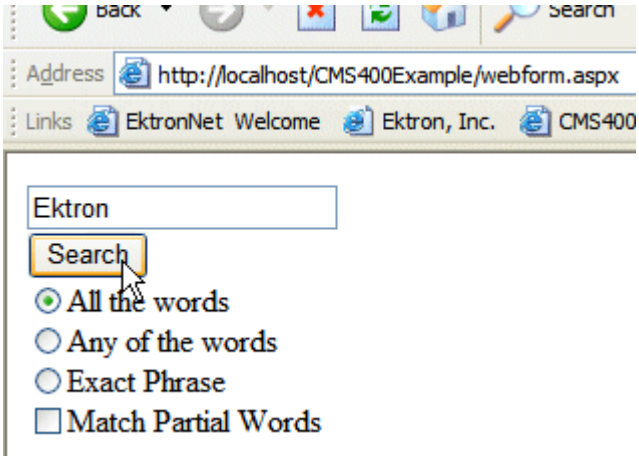
3. Add a second Web form to the project.
4. Drag and drop a Search Server Control on it.



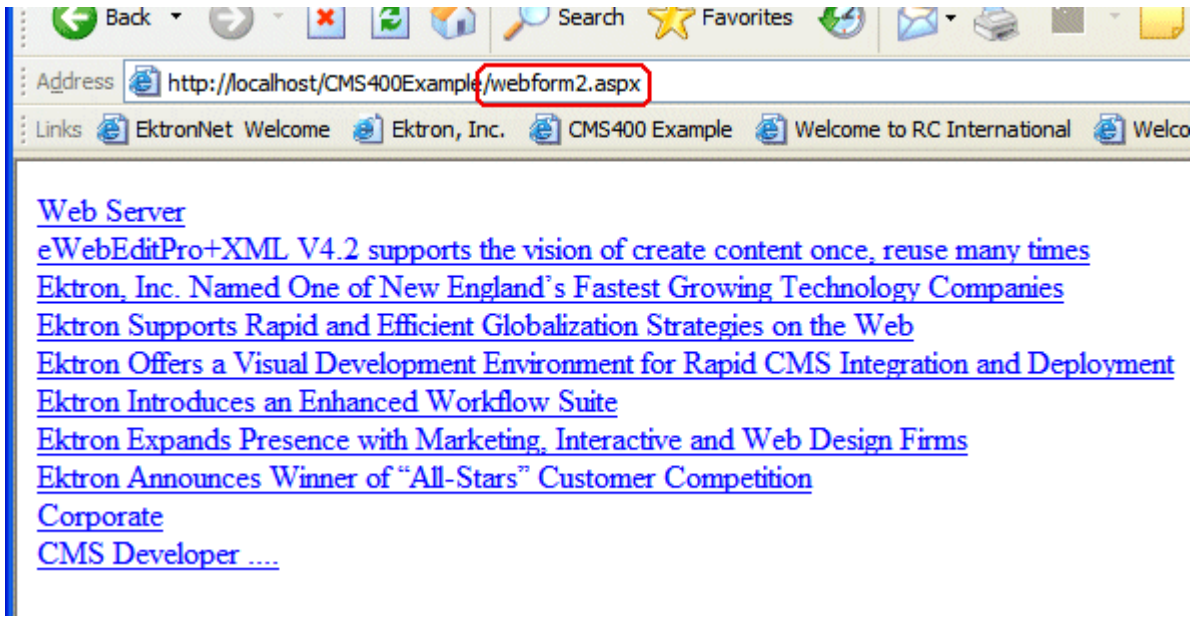

5. Set the Ektron property **ShowSearchBoxAlways** to false.
6. Build the solution.
7. In your browser, go to the first Web form.



8. Use the search dialog.




9. The search results are loaded and displayed in the second Web form automatically.

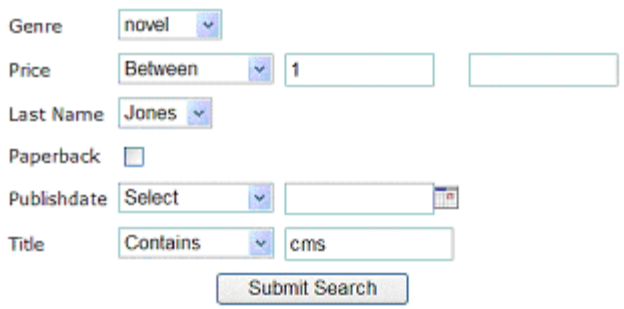

Back
Search
Favorites
Address
http://localhost/CMS400Example/webform2.aspx
Links
EktronNet Welcome
Ektron, Inc.
CMS400 Example
Welcome to RC International
Welco
Web Server
eWebEditPro+XML V4.2 supports the vision of create content once, reuse many times
Ektron, Inc. Named One of New England's Fastest Growing Technology Companies
Ektron Supports Rapid and Efficient Globalization Strategies on the Web
Ektron Offers a Visual Development Environment for Rapid CMS Integration and Deployment
Ektron Introduces an Enhanced Workflow Suite
Ektron Expands Presence with Marketing, Interactive and Web Design Firms
Ektron Announces Winner of "All-Stars" Customer Competition
Corporate
CMS Developer ....

# IndexSearch Server Control

Use the IndexSearch Server Control to display an XML configuration's search screen on any Web form. This lets a user search your site based on XML parameters.

*See Also:* The **Ektron CMS400.NET** Administrator Manual section "XML Indexing".

Below is a sample screen that can be generated via the IndexSearch Server Control.




You can also predefine a search and have the results appear on a Web form template by using the SearchParmXML property.

This subsection contains the following topics:


- "IndexSearch Server Control vs. Search Server Control" on page 98
- "IndexSearch Server Control Properties" on page 98
- "SearchParmXML Property" on page 103
- "Using the IndexSearch Server Control Programmatically Example" on page 106

- ”Programmatically Predefined General Search Results Replaced By Specific Search Results.” on page 109
- ”Turn Off Caching In Netscape Navigator and Mozilla Firefox.” on page 113

# IndexSearch Server Control vs. Search Server Control

See, ”Search Server Control vs. IndexSearch Server Control” on page 86.

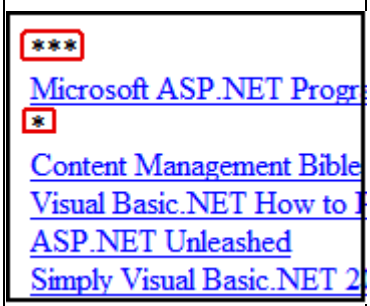
# IndexSearch Server Control Properties

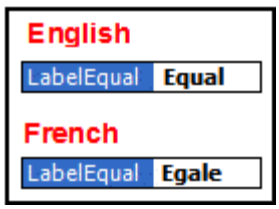
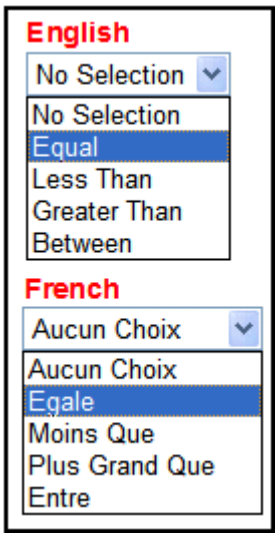
The IndexSearch Server Control properties are described in this table.

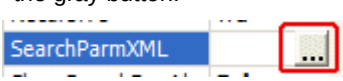
**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Description | Data Type | Values |
|---|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to the folder needed for the **FolderID** property. *See Also:* ”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 32 | String | |
| ButtonText | The text that appears on the button. | String | The default is:<br>Search |

| Property | Description | Data Type | Values |
|---|---|---|---|
| ContentParameter | Checks the QueryString for the content block ID value and replaces the search with content a when ID is specified. Leave blank to always display the search. | String | None - Use Default<br>ID - The server control uses the content block passed as a URL parameter. |
| DisplayXslt | Determines how the information on the page is displayed | String | **None**-databind only<br>**ecmNavigation** - lists the title of every content block in the folder<br>*See Also:* "Example of ecmNavigation Display" on page 122<br>**ecmTeaser** - lists the title of every content block in the folder plus the content summary<br>*See Also:* "Example of ecmTeaser Display" on page 123<br>**Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page |
| EmptyResultMsg | The message that displays if the search returns no hits. | String | The default value is:<br>Your search did not match any documents. |
| FolderId | The numeric id of the folder that you want to search. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32<br>The Recursive property determines whether the IndexSearch Server Control will use this folder's child folders in the search. | Integer | Any folder's numeric ID<br>**0** (zero) represents the root folder. |

| Property | Description | Data Type | Values |
|---|---|---|---|
| Hide | Used to hide output of the index search in design time and run time. For example, you want to pass the results to code behind for further manipulation. | Boolean | **True** = Hide results<br>**False** = show results |
| HitsMarker | This symbol shows how many search criteria each hit matched.<br>The following example shows the first hit met three of the criteria. The second group met one criterion.<br>  <br><br>This property is used when the weighted property is set to true. *See Also:* "Weighted" on page 103. | String | *<br>#<br>+<br>**(Custom)** Custom lets you choose the marker to use. The custom marker can be any text, numeric, or symbol character except **&** and **<**.<br><br>For more Information, see The Ektron CMS400.NET Administrator Manual section "XML Indexing" > "Displaying Search Results" |

| Property | Description | Data Type | Values |
|---|---|---|---|
| Labelafter<br>Labelbefore<br>LabelBetween<br>LabelContains<br>LabeldateBetween<br>LabelEqual<br>LabelExactPhrase<br>LabelGreaterThan<br>Labelinputdate<br>Labelinputnumber<br>LabelLessThan<br>Labelon<br>LabelSelect | All of these properties let you change their label names. This is useful for Multi-language issues. For example, if you have a French site, you change **Equal** to **Égale.**<br>  <br>Note: These label properties will not change based on the **Language** property's setting. For example, setting the Language property to 1036 will not change these labels to French. | String | Examples of what the user will see on the site in French and English.<br>.<br>  |
| Language | Design time only: View content blocks, collections, etc. in different languages.<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer | |
| MaxResults | The Maximum number of content blocks returned (0=unlimited). | Integer | |

| Property | Description | Data Type | Values |
|---|---|---|---|
| OrderBy | The order of the list to be returned. For example, you want to sort search results by last modified date. | String | • **Title** - The title of the content block<br>• **ID** - The XML content block ID number<br>• **Date Created** - The date the content block was created<br>• **Date Modified** - The date the content block was last modified<br>• **AuthorLName** - The last author's last name<br>• **AuthorFname** - The last author's first name<br>• **StartDate** - The GoLive Date |
| OrderByDirection | The direction the results will be sorted in. This property works with the **OrderBy** property only. | String | Ascending<br>Descending |
| Recursive | Determines whether the IndexSearch Server Control will use this folder's child folders in the search. | Boolean | True<br>False |
| SearchParmXML | Opens a separate window that lets you predefine a search. For example, you want a predefined list of books to appear on a web page.<br>To access the window click the gray button.<br>  | | For more information on SearchParmXML, see "SearchParmXML Property" on page 103 |
| ShowSearchBoxAlways | If set to **false**, the search box does not appear on PostBack | Boolean | True<br>False |

| Property | Description | Data Type | Values |
|---|---|---|---|
| Weighted | If you set this value to **true**, content blocks are ranked by the number of criteria matches.<br>For example, if a content block matches all search criteria, it appears at the top of the results page. A hit marker appears for each match.<br>If a second content block only matches one criterion, it also appears on the results page underneath the first content block. One asterisk appears above the page's title to indicate the single match. | Boolean | **True** - display every content block that matches *at least one* search criteria<br>**False** - show only content blocks that match *all* search criteria |
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**. | String | **Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. |
| xmlConfigID | ID of the XML configuration whose indexed fields will appear on the search screen. | Integer | |

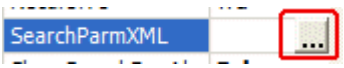
## SearchParmXML Property

The SearchParmXML property is used to create a predefined XML search, whose results appear upon page load. Clicking on the gray box in this property opens a window, where you define the search. The parameters in this window are defined by the xmlConfigID you choose. Using this property launches the CMS Explorer login, if you are not already logged in. For more information on the CMS

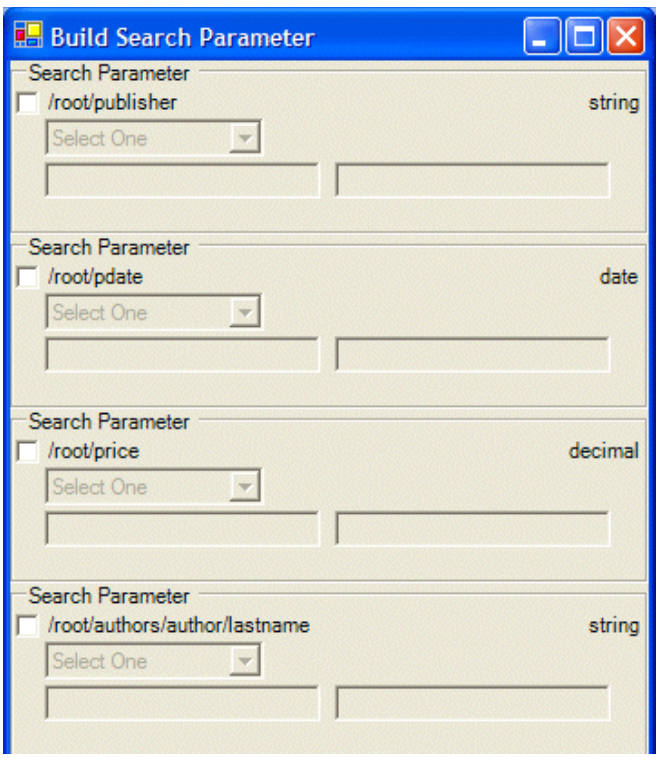
Explorer login, see ”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 32

The following example uses the SearchParmXML property to create a list of books, based on publishers with “press” in their name.

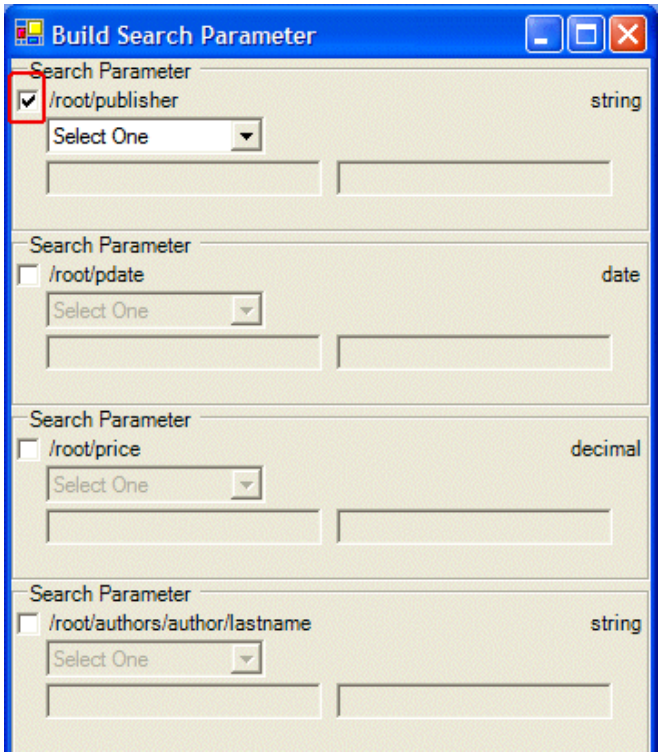
1. To activate the window, click the gray box in the SearchParmXML property.



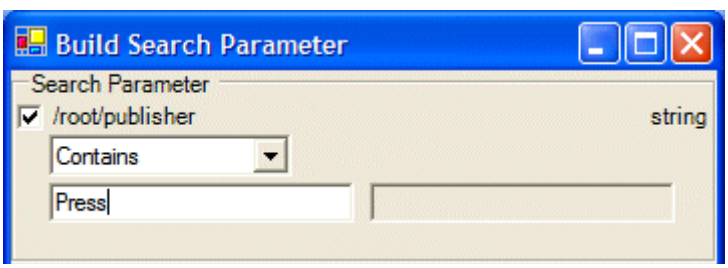

2. The Build Search Parameter screen appears.



3. Click the check boxes for the parameters you want to search with.




4. Define the parameter.



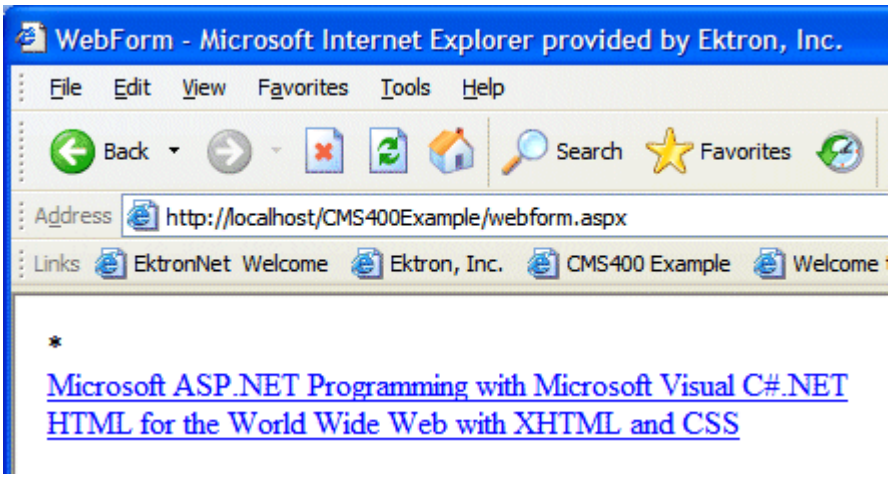

The drop down box in this example contains the following variables:

**Select One** - Prompts you to choose a variable.

**Exact Phrase** - You enter a phrase and search for that exact phrase, contained within the XML content.

**Contains** - You enter a word or phrase and search for any XML content, that has that word or phrase contained within it.

5. Click OK.
6. Build the Web form.
7. Browse to the Web page in your browser.




8. The page appears with a list of XML content blocks that match the parameters you set.

# Using the IndexSearch Server Control Programmatically Example

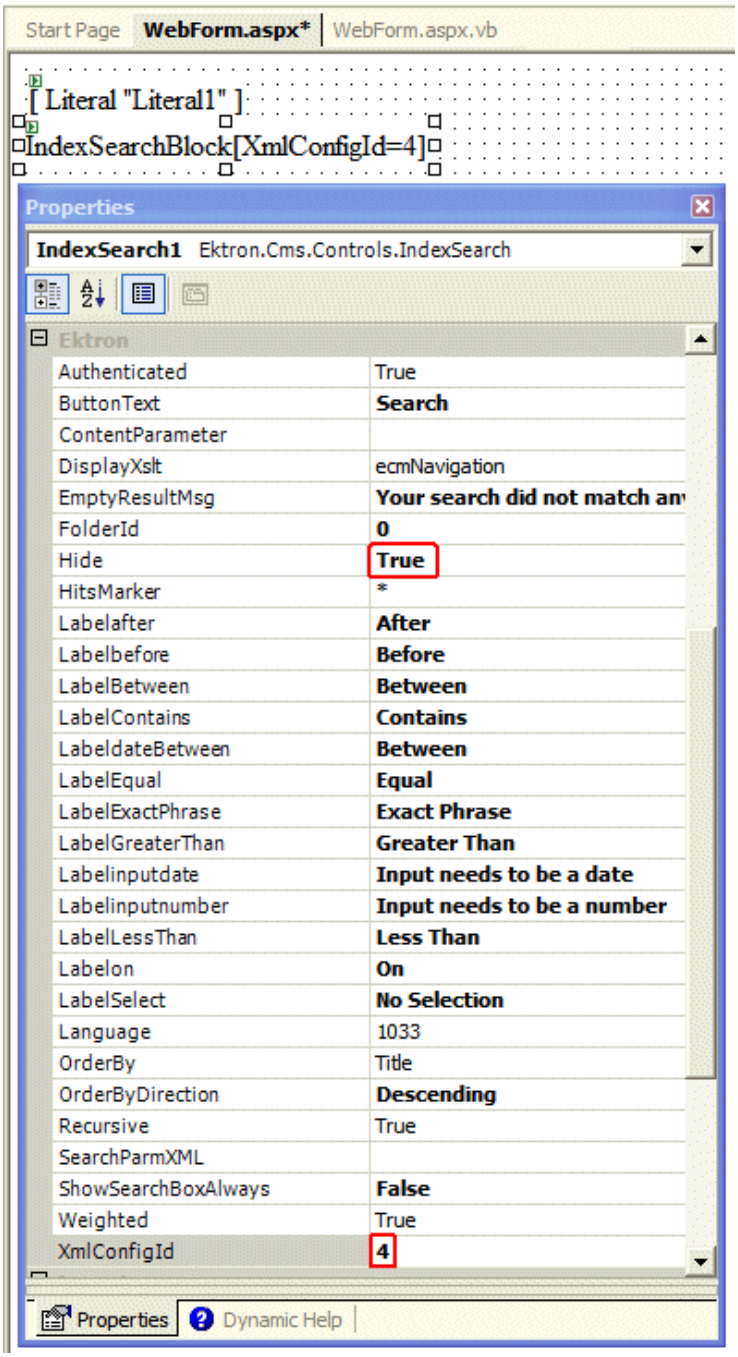
**NOTE** When creating an IndexSearch Server Control form user interface using code behind only, you need the IndexSearch Server Control to be connected with the page events. Otherwise you will not be able to submit your search parameters.

You can use the IndexSearch Server Control, in code behind, to manipulate the way users view the search results. The following example, using code behind, shows how to return values associated with an XPath. This example shows a list of book subject titles.

1. Open a Web form.
2. Drag and drop an IndexSearch Server Control on your Web form.
3. Set the properties in the properties window.

**NOTE** In this example, the hide setting must be set to **true**. Remember to set the XmlConfigID. This example uses ID number **4**.

4. Drag and drop a Literal on the Web form.



5. Add the following code to the code behind.

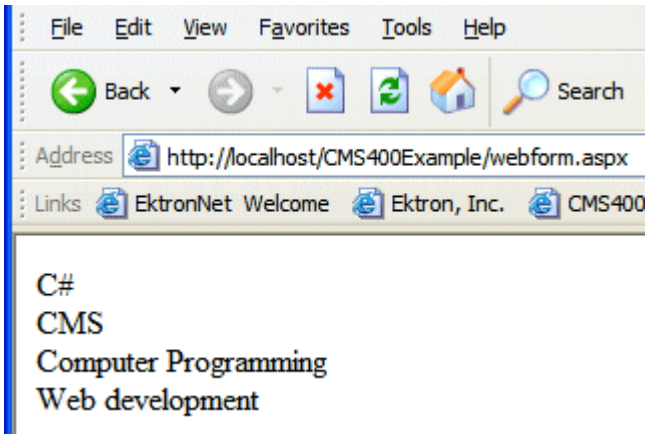
```
IndexSearch1.Parm.XPath = "/root/subject"

Dim arStr As String()

Dim Str As String

Dim strRet As String

arStr = IndexSearch1.GetXPathValues("/root/subject")

For Each Str In arStr

    strRet += Str & "<br/>"

Next

Literal1.Text = strRet
```

Here is a description of what is happening in the code.

- This line defines the parameter XPath location:
  `IndexSearch1.Parm.XPath = "/root/subject"`
- The next three lines create variables:
  ```
  Dim arStr As String()
  Dim Str As String
  Dim strRet As String
  ```
- This line set the variable arStr as an array of the XPathValues:
  ```
  arStr = IndexSearch1.GetXPathValues("/root/
  subject")
  ```
- The for next loop iterates through the array of XPathValues:

  ```
  For Each Str In arStr
      strRet += Str & "<br/>"
  Next
  ```
- The last line displays the values on your web form:
  `Literal1.Text = strRet`

6. Build the project.
7. Browse to your web form.
8. The list of book subject titles appears.




# Programmatically Predefined General Search Results Replaced By Specific Search Results.

This Code behind example shows a predefined search appearing on a Web page, then being replaced by more specific search results. This example creates a hyperlink list of book publishers. When you click a specific publisher, it replaces the publisher list with a book list from that publisher.

Follow these steps.

1. Add an IndexSearch Server Control to a Web form.
2. Set the following properties:

- DisplayXslt = ecmTeaser
- Hide = True
- HitsMarker = (blank)
- ShowSearchBoxAlways = False
- Weighted = True
- XmlConfigId = 4

3. Add two Literals.
4. Add the following code to the Page Load Event in code behind.

```
Dim bRet As Boolean
   If Request.QueryString("publisher") <> "" Then
```

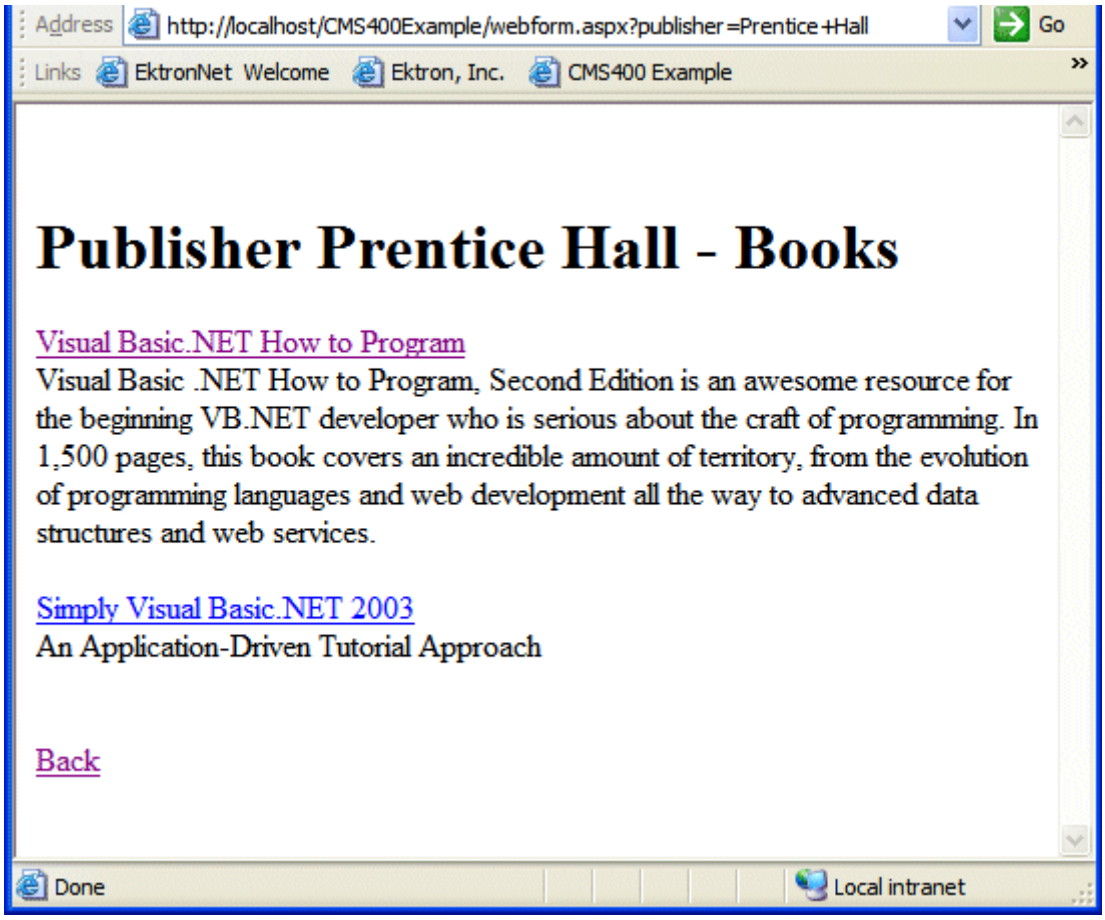
```
      IndexSearch1.Hide = False
      IndexSearch1.Parm.XPath = "/root/publisher"
      IndexSearch1.Parm.DataType =
Ektron.Cms.Common.EkSearch.XMLDataType.String
      IndexSearch1.Parm.SearchType =
Ektron.Cms.Common.EkSearch.XMLSearchRangeType.ExactPhrase
      IndexSearch1.Parm.Value1 = Request.QueryString("publisher")
      bRet = IndexSearch1.AddParm()
      IndexSearch1.Search()

      Literal1.Text = "<h1>Publisher " & Request.QueryString("publisher") &
" - Books</h1>"
      Literal2.Text = "<a href=""webform.aspx"">Back</a>"
   Else
      Literal1.Text = "<h1>Publishers</h1>"
      Dim arStr As String()
      Dim Str As String
      Dim strRet As String
      arStr = IndexSearch1.GetXPathValues("/root/publisher")
      For Each Str In arStr
        strRet += "<a href=""webform.aspx?publisher=" &
Server.UrlEncode(Str) & """>" & Str & "</a><br/>"
      Next
      Literal1.Text += strRet
   End If
```

5. Save your Web form.
6. Build your Web form.
7. Browse to the Web form in your browser.

8. The list of publishers appears.

9. Click on a publisher.

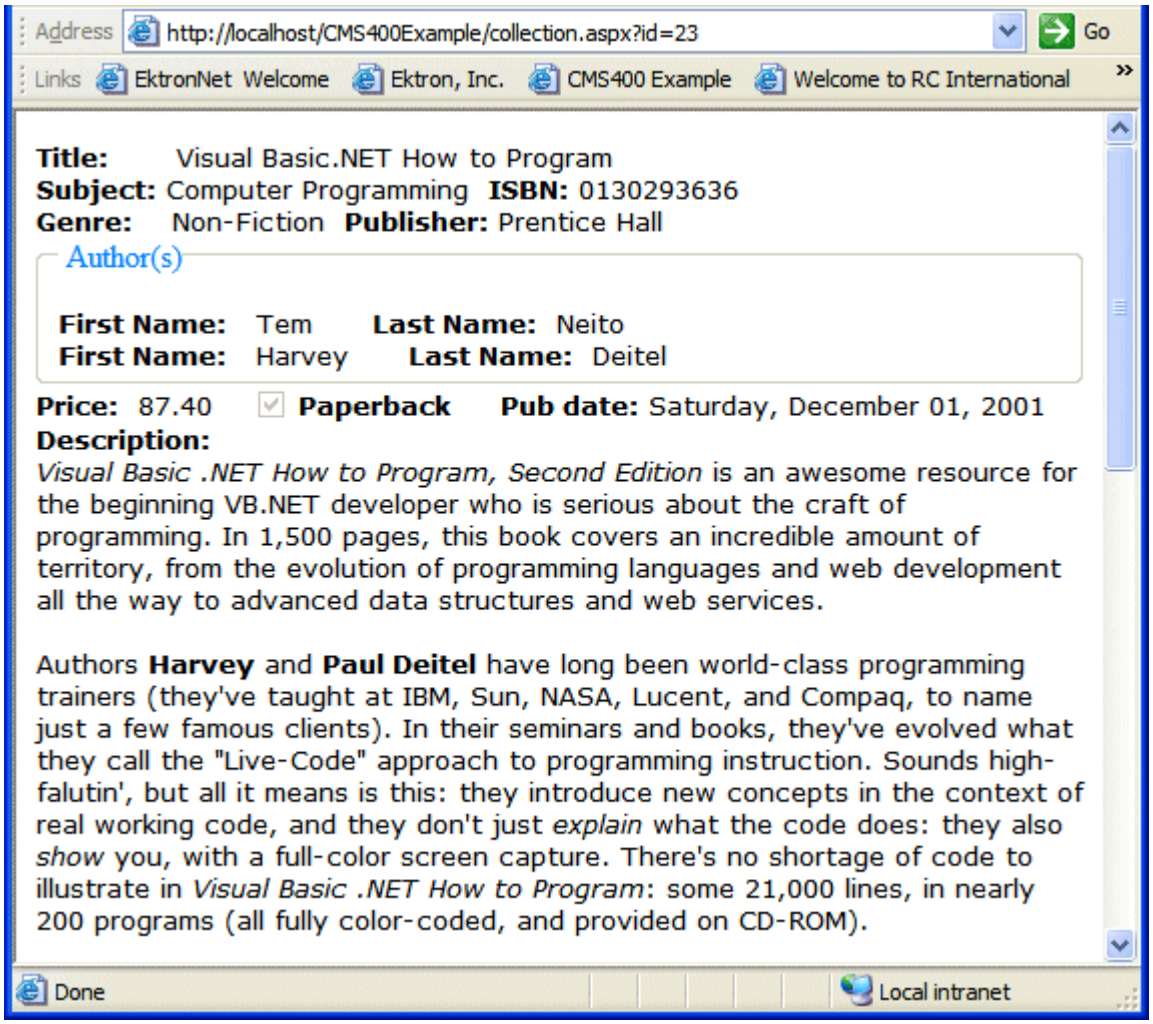
10. A list of books with teasers by that publisher appears.


Address
http://localhost/CMS400Example/webform.aspx?publisher=Prentice+Hall
Go
Links
EktronNet Welcome
Ektron, Inc.
CMS400 Example
Publisher Prentice Hall - Books
Visual Basic.NET How to Program
Visual Basic .NET How to Program, Second Edition is an awesome resource for the beginning VB.NET developer who is serious about the craft of programming. In 1,500 pages, this book covers an incredible amount of territory, from the evolution of programming languages and web development all the way to advanced data structures and web services.
Simply Visual Basic.NET 2003
An Application-Driven Tutorial Approach
Back
Done
Local intranet

Notice the second Literal now contains a hyperlink to go back.

11. Click on a book.

12. The XML Content Block for that book appears.


Address
http://localhost/CMS400Example/collection.aspx?id=23
Go
Links
EktronNet Welcome
Ektron, Inc.
CMS400 Example
Welcome to RC International
Title: Visual Basic.NET How to Program
Subject: Computer Programming ISBN: 0130293636
Genre: Non-Fiction Publisher: Prentice Hall
Author(s)
First Name: Tem Last Name: Neito
First Name: Harvey Last Name: Deitel
Price: 87.40 Paperback Pub date: Saturday, December 01, 2001
Description:
Visual Basic .NET How to Program, Second Edition is an awesome resource for the beginning VB.NET developer who is serious about the craft of programming. In 1,500 pages, this book covers an incredible amount of territory, from the evolution of programming languages and web development all the way to advanced data structures and web services.
Authors Harvey and Paul Deitel have long been world-class programming trainers (they've taught at IBM, Sun, NASA, Lucent, and Compaq, to name just a few famous clients). In their seminars and books, they've evolved what they call the "Live-Code" approach to programming instruction. Sounds high-falutin', but all it means is this: they introduce new concepts in the context of real working code, and they don't just explain what the code does: they also show you, with a full-color screen capture. There's no shortage of code to illustrate in Visual Basic .NET How to Program: some 21,000 lines, in nearly 200 programs (all fully color-coded, and provided on CD-ROM).
Done
Local intranet

# Turn Off Caching In Netscape Navigator and Mozilla Firefox.

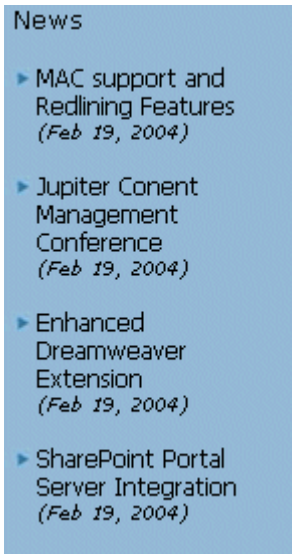
When using Netscape Navigator or Mozilla Firefox to perform an index search some parameters become cached. To prevent caching in Netscape Navigator and Mozilla Firefox, add the following line to the Page Load Event in code behind:

```
Response.Cache.SetNoStore()
```

# ListSummary Server Control

The List Summary Server Control is used to display a list of content blocks on a Web page.

When added to a template and visited, a List Summary looks like this.




It displays each content block's title and summary information. You can modify the display to suit your needs by modifying its properties. The List Summary Server Control properties are described in this table.

This subsection contains the following topics:


- "ListSummary Server Control Properties" on page 115
- "List Summary Example" on page 119
- "OrderKey Property" on page 118

# ListSummary Server Control Properties

The ListSummary Server Control properties are described in this table.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

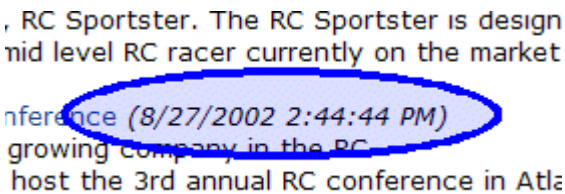
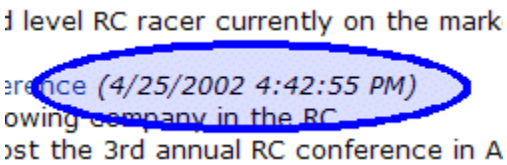
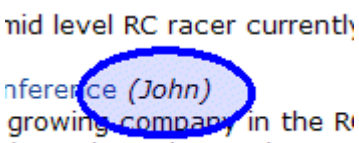
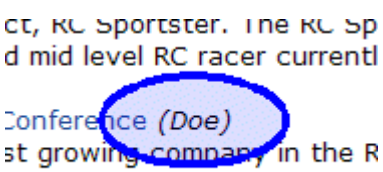
| Property | Description | Data Type | Values |
|---|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to the folder needed for the **FolderPath** property. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | String | |
| ContentParameter | Checks the QueryString for this value and replaces the collection with a content block when specified. Leave blank to always display the Collection or List Summary. | String | |
| Direction | How to order the hyperlinks on the list.<br>The sort field is determined by the OrderKey property. | either ascending or descending | ascending - they are arranged A, B, C or 1,2,3.<br>descending - they are arranged. Z,Y,X or 3,2,1<br>If sorting by date, descending puts the most recent first. |

| Property | Description | Data Type | Values |
|---|---|---|---|
| DisplayXslt | Determines how the information on the page is displayed | String | **None**-databind only<br>**ecmNavigation** - lists the title of every content block in the folder<br>*See Also:* "Example of ecmNavigation Display" on page 122<br>**ecmTeaser** - lists the title of every content block in the folder plus the content summary<br>*See Also:* "Example of ecmTeaser Display" on page 123<br>**Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page |
| FolderPath | The folder that contains the content blocks which appear in the List Summary. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32<br>The Recursive property determines whether content blocks in this folder's child folders also appear. | String | Any folder |
| Hide | Used to hide output of List Summary in design time and run time. | Boolean | **True** = Hide results<br>**False** = show results |
| Language | Design time only: View content blocks, collections, etc. in different languages.<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer | Integer |

| Property | Description | Data Type | Values |
|---|---|---|---|
| MaxResults | Maximum number of summaries displayed on a Web page. | Integer | any integer<br>0 = unlimited |
| OrderKey | Sort the list by one of the values. | must be one of the values | Title<br>DateModified<br>DateCreated<br>LastEditorFname<br>LastEditorLname<br>*See Also:* "OrderKey Property" on page 118 |
| Random | Set to **True** if you want to randomly display one content block in the specified folder. The content changes each time a user views the page. | Boolean | True<br>False |
| Recursive | Determines if the display includes content blocks in child folders of the selected folder. | Boolean | True<br>False |
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**. | String | **Span** - The  tag is used to designate an inline portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. |

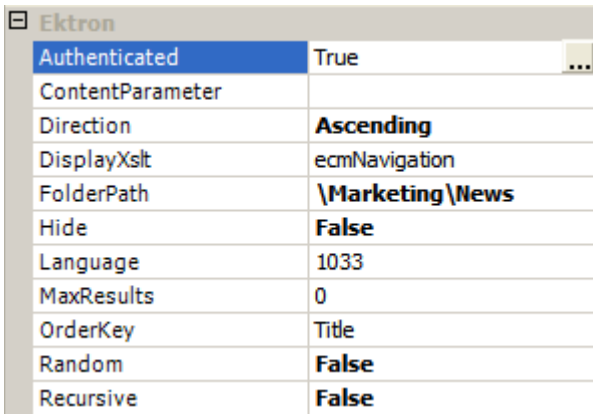
## OrderKey Property

The OrderKey property determines what information appears to the right of the hyperlinked title. The values are described below.

| Value | Description | Web Page View |
|---|---|---|
| DateModified | Displays the date the content was last modified. |  , RC Sportster. The RC Sportster is design<br>mid level RC racer currently on the market<br>nference (8/27/2002 2:44:44 PM)<br>growing company in the RC<br>host the 3rd annual RC conference in Atla |
| DateCreated | Displays the date when the content was first created. |  d level RC racer currently on the mark<br>erence (4/25/2002 4:42:55 PM)<br>owing company in the RC<br>ost the 3rd annual RC conference in A |
| LastEditorFname | Displays the first name of the user who last edited the content block. |   |
| LastEditorLnames | Displays the last name of the user who last edited the content block. |  ct, RC Sportster. The RC Sp<br>d mid level RC racer currentl<br>Conference (Doe)<br>st growing company in the R<br>to host the 3rd annual RC c |

# List Summary Example

Below is an example of the List Summary Server Control properties.



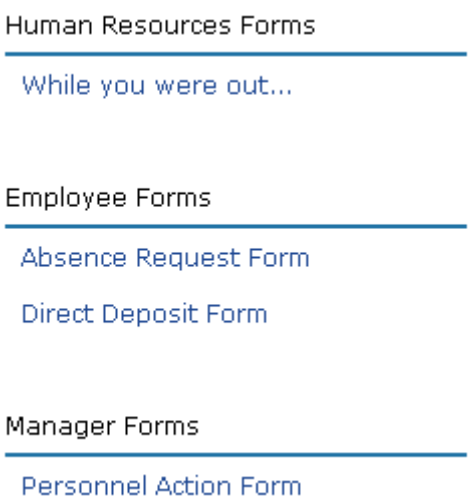

The example has the following properties:

- Direction - The results are displayed in ascending order by title
- FolderPath - Displays summaries from the `\Marketing\News` folder
- MaxResults - Unlimited results appear
- OrderKey - Summaries sorted by content block title
- Random - Displays summary for all content blocks in folder, not just one random one
- Recursive - Includes only selected folder, not its child folders
- Xslt - will use the navigation display format

# Collection Server Control

The Collection Server Control is used to display a list of content block links that were created as a collection in the Ektron CMS400.NET Workarea. The control is highly customizable, allowing you to define the display of the collection on a Web page.

When added to a template and visited, collections can look like this (The following illustration shows three collections on the same page of a sample site.) You can modify the display by editing its properties.




This subsection contains the following topics:


- ”Collection Server Control Properties” on page 121
- ”Example of ecmNavigation Display” on page 122
- ”Example of ecmTeaser Display” on page 123
- ”Using the Collection Server Control Programmatically Example” on page 124

# Collection Server Control Properties

The Collection Server Control properties are described in this table.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Description | Data Type |
|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | String |
| ContentParameter | Checks the QueryString for this value and replaces the collection with a content block when specified. Leave blank to always display the Collection or List Summary. | String |
| DefaultCollectionID | The ID of a collection that appears where you inserted this server control if no other collection is identified, or is not available.<br>If you don't know the ID number of the collection, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | Integer |
| DisplayXslt | Determines how the information on the page is displayed<br>**None**-databind only<br>**ecmNavigation** - lists the title of every content block in the collection<br>*See Also:* "Example of ecmNavigation Display" on page 122<br>**ecmTeaser** - lists the title of every content block in the collection plus the content summary<br>*See Also:* "Example of ecmTeaser Display" on page 123<br>**Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page | String |

| Property | Description | Data Type |
|---|---|---|
| DynamicParameter | To make this collection dynamic, select **coll_id**. When you do, this server control uses the collection passed as a URL parameter. | String |
| GetHtml | Set to **True** if you want to retrieve and display content (html body) for all content blocks in the collection. For example, to display content inside a web server control such as a DataGrid. | Boolean |
| Hide | Used to hide output of collection in design time and run time.<br>**True** = Hide collection<br>**False** = Display collection | Boolean |
| Language | Design time only: View content blocks, collections, etc. in different languages.<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |
| Random | Set to **True** if you want to randomly display one collection item. The item changes each time a user views the page. | Boolean |
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an inline portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |

## Example of ecmNavigation Display

The following is an example of a collection being used as a navigation menu.

### *ecmNavigation XSL code*

Below is the XSL code that is used to create ecmNavigation Display. You can use this code as the basis to design your own XSLT.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/
XSL/Transform">

<xsl:template match="/">

<table border="0" cellspacing="0" cellpadding="0" width="100%">

<xsl:for-each select="Collection/Content">

<tr><td><a><xsl:attribute name="href">

<xsl:value-of select="QuickLink"/></xsl:attribute><xsl:value-of
select="Title"/>

</a></td></tr></xsl:for-each></table></xsl:template></
xsl:stylesheet>
```

## Example of ecmTeaser Display

The following is an example of a collection using the ecmTeaser display style.

Home | Products | Support | Latest News | Careers | Search | Calendar
Employment Opportunities
Plastic Molder #123
RC International is looking for an experienced plastics molder.
Servo Control Engineer #124
RC International is looked for a highly skilled servo control engineer to join our team.

### *ecmTeaser XSL code*

Below is the XSL code that is used to create ecmTeaser Display. You can use this code as the basis to design your own XSLT.

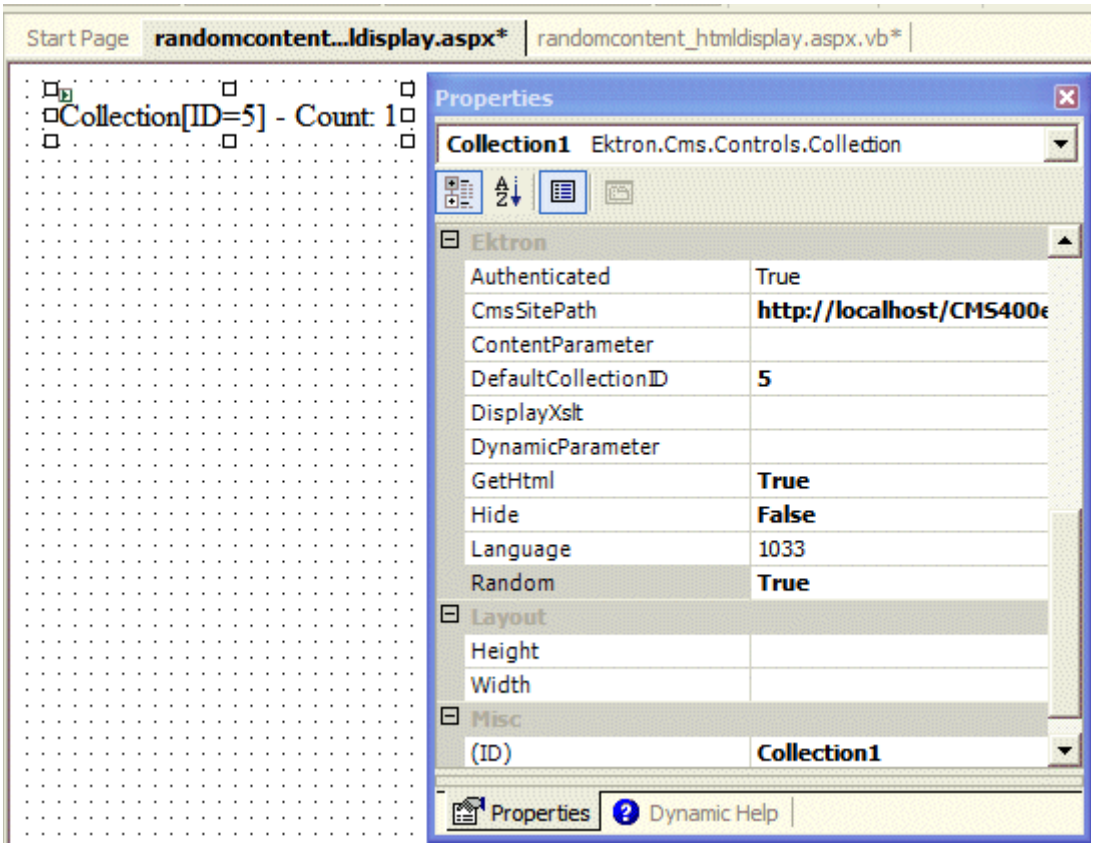
```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/
XSL/Transform">

<xsl:template match="/"><table border="0" cellspacing="0"
cellpadding="0" width="100%">

<xsl:for-each select="Collection/Content"><tr><td><a>

<xsl:attribute name="href"><xsl:value-of select="QuickLink"/></
xsl:attribute>

<xsl:value-of select="Title"/></a> 

   // xml.Append("<xsl:value-of
select="substring(DisplayDateModified, 0, string-
length(DisplayDateModified)-11)" />

</td></tr><tr><td><xsl:value-of select="Teaser" /></td></
tr><tr><td> </td></tr>

</xsl:for-each></table></xsl:template></xsl:stylesheet>
```

# Using the Collection Server Control Programmatically Example

The following is an example of using code behind, and a drag and drop Collection Server Control to display random content from a collection.

1. Drag and Drop a Collection Server Control on your Web form.
2. Set the properties in the properties window.

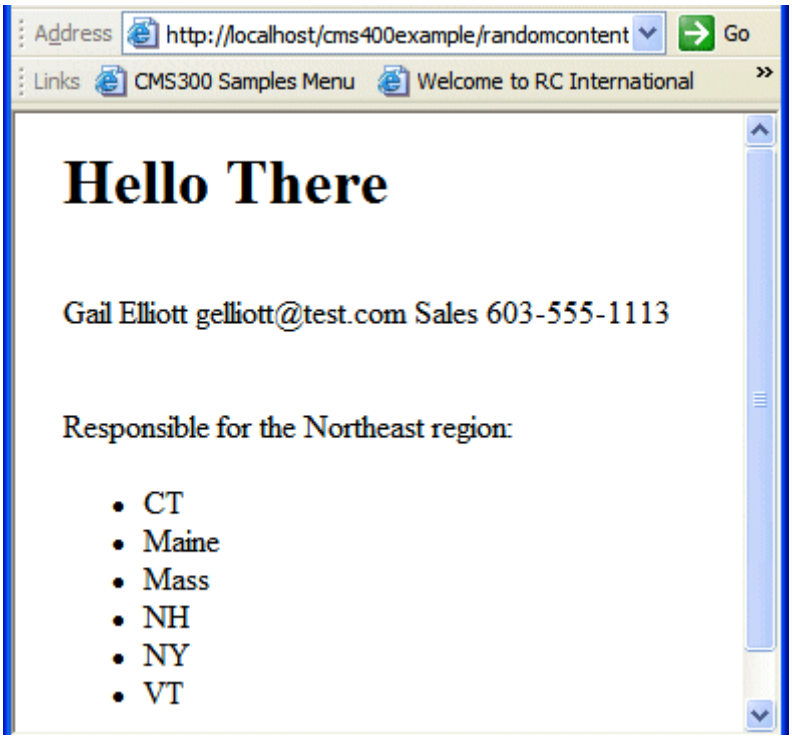
**NOTE** In this example, the Random property and the GetHtml property must be set to **True**.




3. Add the following code to the code behind.

```
Dim str As String

str = "<h1>Hello There</h1><br>"

str &= Collection1.EkItems(0).Html

Collection1.Text = str
```

4. Build and browse your web form.


Address
http://localhost/cms400example/randomcontent
Go
Links
CMS300 Samples Menu
Welcome to RC International
Hello There
Gail Elliott gelliott@test.com Sales 603-555-1113
Responsible for the Northeast region:
CT
Maine
Mass
NH
NY
VT

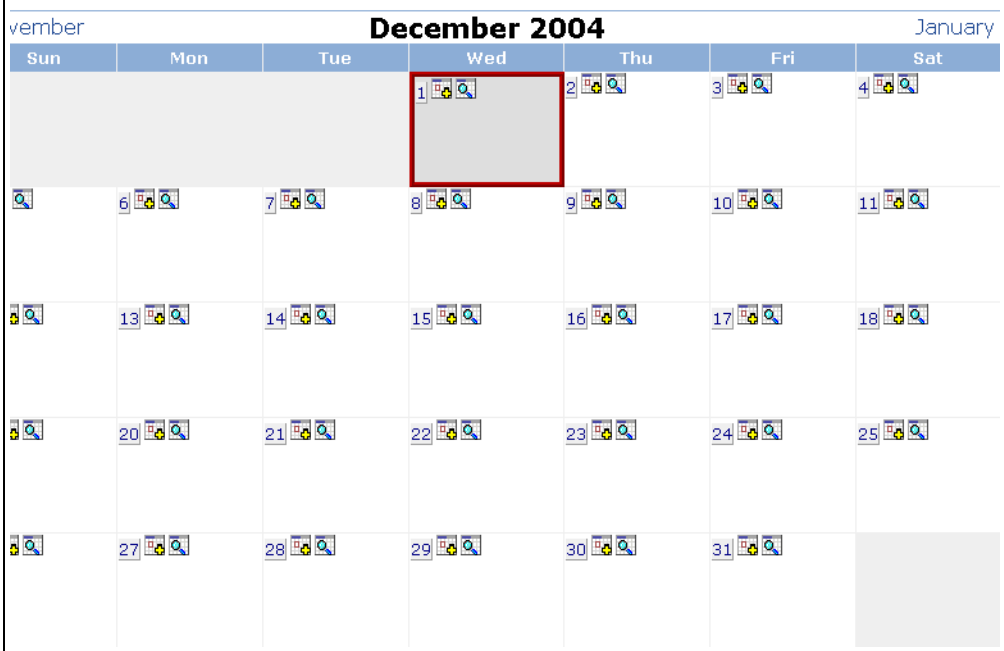
# Calendar Server Control

The Calendar Server Control displays a calendar that was created in the Workarea. By using the Calendar Server Control's properties, you can select multiple views of the calendar to display.
*See Also:* "DisplayType" on page 131

You can adjust a calendar's look and feel by using a style sheet.
*See Also:* "Calendar Style Sheet" on page 133 and "calendarStyles.css Class Description" on page 133.

The Calendar Server Control can output the raw XML data of the calendar. You can use this XML data to create a completely customized calendar.

Below is a Calendar displayed on the Web Page. When logged in, you can add and view calendar events. When logged out, calendar events only appear.

| Login Status | Calendar display |
| --- | --- |
| Logged in |  |

| Login Status | Calendar display |
|---|---|
| Logged Out | |



A calendar can be *static* or *dynamic*.

- A static calendar is selected in the server control properties.
- A dynamic calendar is passed to the page by a URL parameter.

This subsection contains the following topics:


- ”Calendar Server Control Properties” on page 129
- ”Calendar Style Sheet” on page 133
- ”calendarStyles.css Class Description” on page 133


# Calendar Server Control Properties

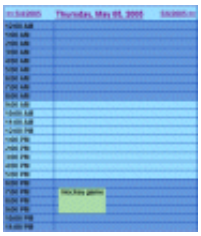
The Calendar Server Control properties are described in this table.

**NOTE** There are additional settings to help define the calendar in the Ektron CMS400.NET Workarea. These include display weekends, forward only, show all

label, etc. For information on these properties, see Ektron CMS400.NET **Administrator's Manual** > **Calendars** > **Managing Calendar Modules** > **Viewing Calendar Modules**.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | String |
| DefaultCalendarID | The ID number of the calendar that appears where you inserted this server control.<br>If you don't know the ID number of the calendar, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | Integer |

| Property | Value | Data Type |
|---|---|---|
| DisplayType | Specify the Calendar view type. The default is **Month**.<br>• **Month** - A 30 day calendar month view.<br> <br>• **Day** - Single day view.<br>  <br>• **Inline** - Display events that occur from the start date of the calendar through the end date.<br> <br>• **MonthUpWithEvent** - Displays a month calendar. Events appear to the right of the calendar when you click on a day.<br> | String |
| DynamicParameter | To make this calendar dynamic, select **calendar_id**. When you do, this server control uses the calendar passed as a URL parameter. | String |

| Property | Value | Data Type |
|---|---|---|
| EndDate | Specify the end date the calendar displays. Used in conjunction with the StartDate property. For example, you have a calendar that shows only classes for the next three month semester. | Date |
| Hide | Used to hide output of calendar in design time and run time.<br>**True** = Hide calendar<br>**False** = Display calendar | Boolean |
| Language | Design time only: View the calendar in selected languages.<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |
| ShowXML | At runtime, output the raw XML data instead of the calendar. With this XML data, you can apply your own XSL to create your own custom display formats.<br>Default is **False.**<br>**True** = Show raw XML data<br>**False** = Show Calendar | Boolean |
| StartDate | Specify the start date the calendar displays. Used in conjunction with the EndDate property. For example, you have a calendar that shows only classes for the next three month semester. | Date |
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an inline portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |

# Calendar Style Sheet

Before you insert a Calendar into your Ektron CMS400.NET template, Ektron recommends referencing the style sheet `\webroot\cms400example\calendarStyles.css` in your template. Copy the style sheet to your webroot folder then add it to your project. Next create a reference to calenderStyles.css. For example:

```
html>

 <head>

   <title>Company Calendar</title>

    <link href="calendarStyles.css" rel="stylesheet" type="text/
css" />

 </head>
```

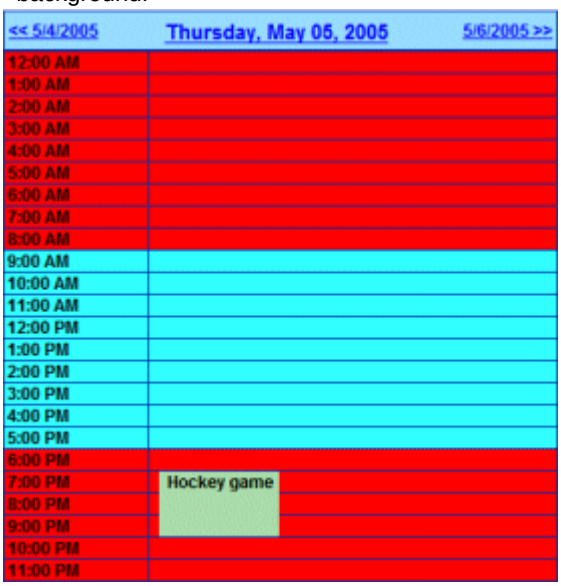
This can also be accomplished via drag and drop. If you drag and drop the calendarStyles.css from the Solution Explorer to your Web form, the reference is created in the HTML automatically.

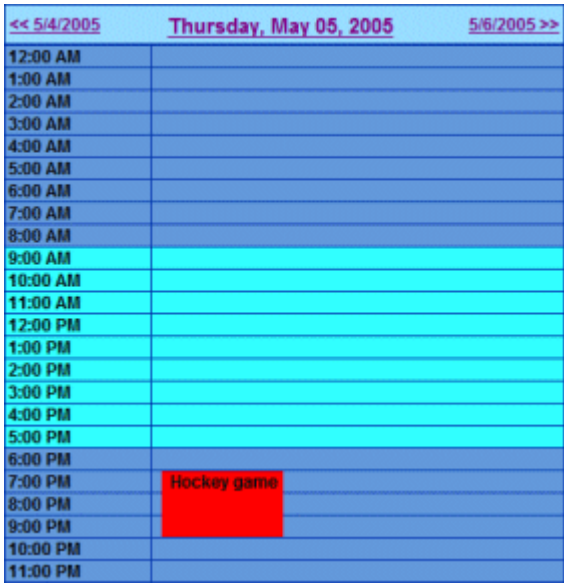
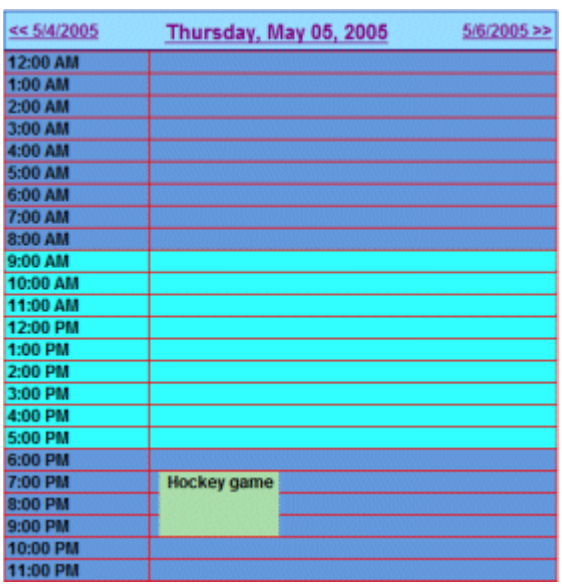
## calendarStyles.css Class Description

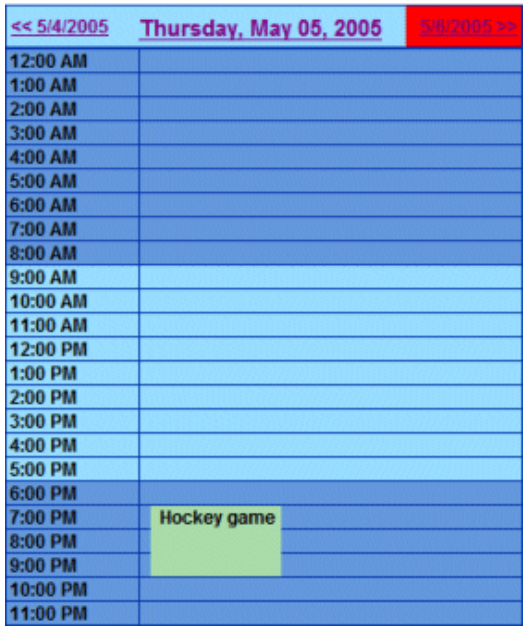
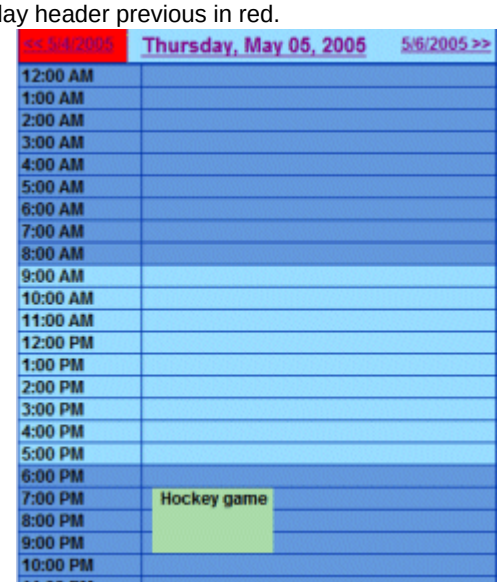
The following table describes the classes found in the calendarStyles.css file. This table contains the following sections.

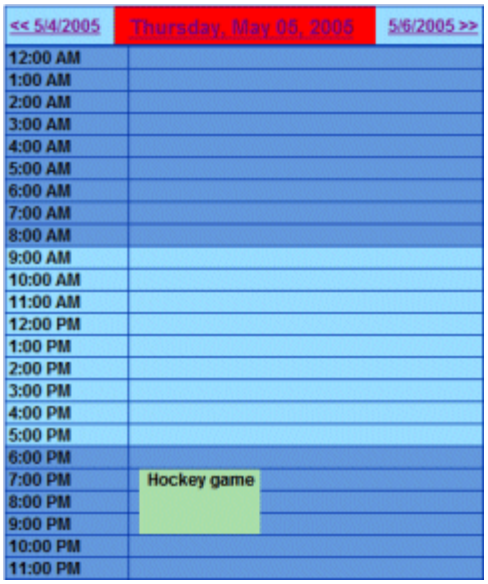

- ”Single Day View (Day)” on page 133
- ”Month View (month)” on page 137
- ”Inline View (inline)” on page 149
- ”Month-Up with Event (monthupwithevent)” on page 151
- ”Event Type Control (used in month display)” on page 158


| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| ***Single Day View (Day)*** | | |

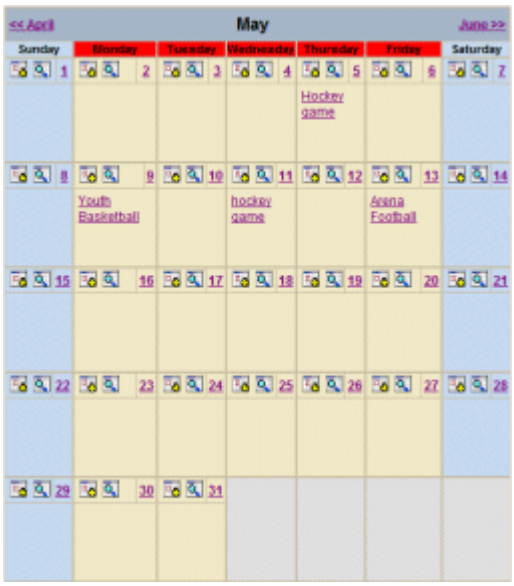
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .dv_HourCell | The workday hour cells. The following example shows these cells with a red background.<br> | • **background**: #99DDFF<br>• **font-family**: Arial, Arial Narrow, Times New Roman<br>• **font-size**: 9pt<br>• **font-weight**: bold |
| .dv_HourCellEve | The evening and morning hour cells. The following example shows these cells with a red background.<br>  | • **background**: #6699DD<br>• **font-family**: Arial, Arial Narrow, Times New Roman<br>• **font-size**: 9pt<br>• **font-weight**: bold |

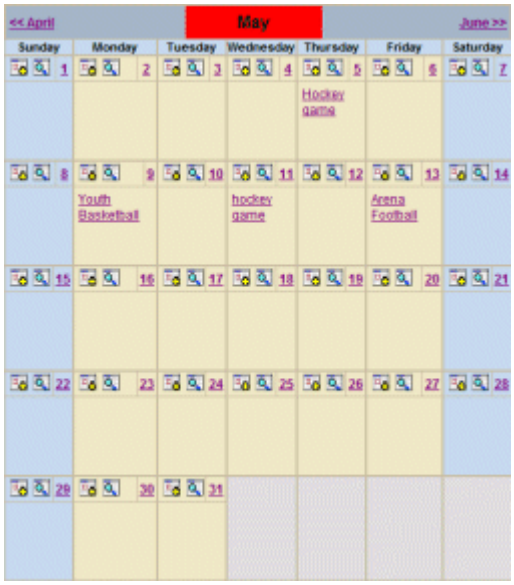
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .dv_EventTitle | An event listed on the daily calendar. The following example shows the event with a red background.<br>  | • **background**: #aaddaa<br>• **font-family**: Arial, Arial Narrow, Times New Roman<br>• **font-size**: 9pt<br>• **font-weight**: bold<br>• **cursor**: pointer |
| .dv_BorderCell | The borders of the hour cells. The following example shows the borders in red.<br>  | • **background**: #0033AA |

| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .dv_DayHeaderNext | The button for moving the calendar to the next day. The following example shows the day header next in red.<br>  | • **background**: #99DDFF<br>• **color**: #0033AA<br>• **font-family**: Arial<br>• **font-size**: 9pt<br>• **font-weight**: bold<br>• **text-align**: right |
| .dv_DayHeaderPrev | The button for moving the calendar to the previous day. The following example shows the day header previous in red.<br>  | • **background**: #99DDFF<br>• **color**: #0033AA<br>• **font-family**: Arial<br>• **font-size**: 9pt<br>• **font-weight**: bold<br>• **text-align**: right |

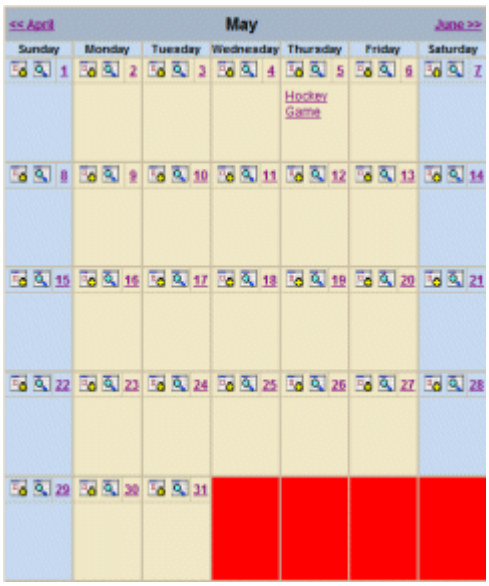
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .dv_DayHeader | The day header for the calendar. The following example shows the day header in red.<br>  | • **background**: #99DDFF<br>• **color**: #0033AA<br>• **font-family**: Arial<br>• **font-size**: 11pt<br>• **font-weight**: bold<br>• **text-align**: center |
| ***Month View (month)*** | | |

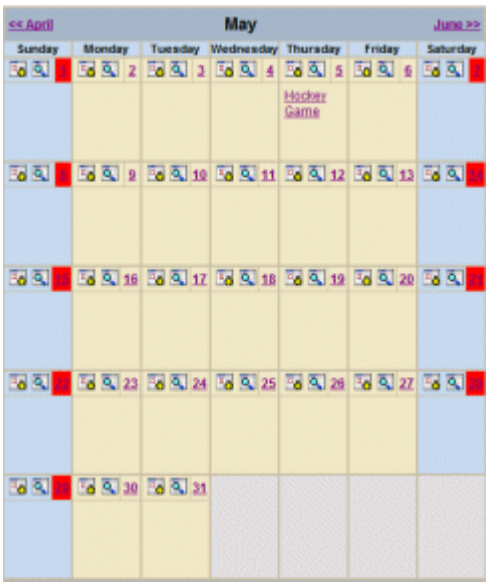
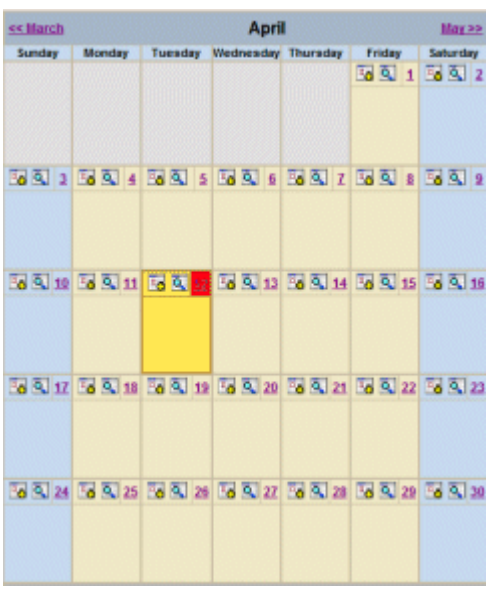
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_weekendDOWHeader | Weekend day of week header. The following example shows the weekend day of week header in red.<br> <br> | • **background**: #c5daef<br>• **font-family**: arial<br>• **font-weight**: bold<br>• **font-size**: 8pt<br>• **border-top**: #c9bda5 1px solid<br>• **border-bottom**: #c9bda5 1px solid<br>• **border-right**: #c9bda5 1px solid<br>• **border-left**: #c9bda5 1px solid<br>• **text-align**: center |

| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_weekdayDOWHeader | Weekday day of week header. The following example shows the weekday day of week header in red.<br>  <br> | • **background**: #c5daef<br>• **font-family**: arial<br>• **font-weight**: bold<br>• **font-size**: 8pt<br>• **border-top**: #c9bda5 1px solid<br>• **border-bottom**: #c9bda5 1px solid<br>• **border-right**: #c9bda5 1px solid<br>• **border-left**: #c9bda5 1px solid<br>• **text-align**: center |

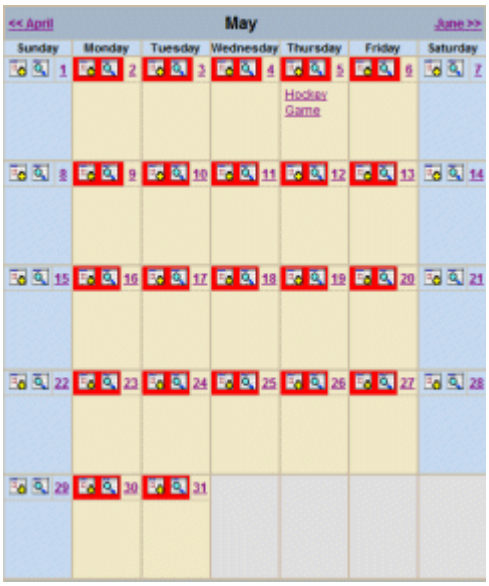
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_MonthHeader | Month header. The following example shows the month header background in red.<br>  | • **font-weight**: bold<br>• **font-size**: 12pt<br>• **background**: transparent<br>• **font-family**: Arial |
| .mv_MonthHeaderBkg | Month header background. The following example shows the month header background in red.<br> | • **border-right**: #c9bda5 2px solid<br>• **border-top**: #c9bda5 2px solid<br>• **border-left**: #c9bda5 2px solid<br>• **font-weight**: bold<br>• **font-size**: 9pt<br>• **background**: #A5B7C9<br>• **font-family**: arial |

| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_MonthHeaderNext | The button for moving the calendar to the next month. The following example shows the month header next background in red.<br> | • **font-weight**: bold<br>• **fontsize**: 9pt<br>• **background**: transparent<br>• **font-family**: Arial<br>• **text-align**: right |
| .mv_MonthHeaderPrev | The button for moving the calendar to the previous month. The following example shows the month header previous background in red.<br> | • **font-weight**: bold<br>• **fontsize**: 9pt<br>• **background**: transparent<br>• **font-family**: Arial<br>• **text-align**: left |

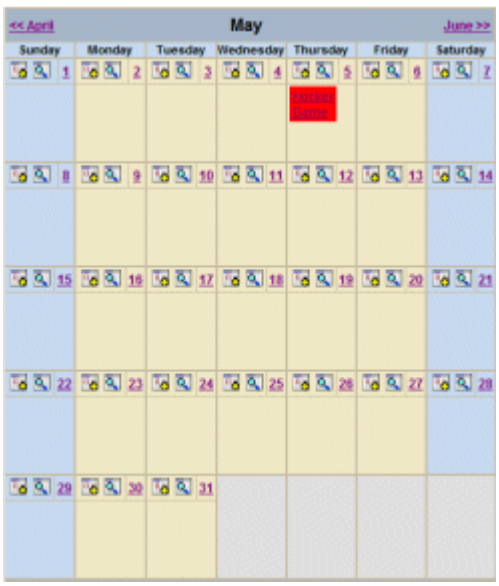
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_NonMonthBorder | The cells at the beginning or end of the month that do not contain days. The following example shows the non month border in red.<br>  | • **border-right**: #c9bda5 1px solid<br>• **border-top**: #c9bda5 1px solid<br>• **background**: #e0e0e0<br>• **border-left**: #c9bda5 1px solid<br>• **border-bottom**: #c9bda5 1px solid |
| .mv_DateCell | The cells where the numerical weekdays appear. The following example shows the date cell in red.<br> | • **font-weight**: bold<br>• **font-size**: 9pt<br>• **background**: #F0E9C5<br>• **border-left**: #c9bda5 1px solid<br>• **border-bottom**: #c9bda5 1px solid<br>• **font-family**: arial |

| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_DateCellWeekend | The cells where the numerical weekend days appear. The following example shows the date cell weekend in red.<br>  | • **font-weight**: bold<br>• **font-size**: 9pt<br>• **background**: #C5DAEF<br>• **border-left**: #c9bda5 1px solid<br>• **border-bottom**: #c9bda5 1px solid<br>• **font-family**: arial |
| .mv_DateCellToday | The cells where today's numerical day appears. The following example shows the date cell today in red.<br>  | • **background**: #FFE354<br>• **border-left**: #b38b3b 1px solid<br>• **border-bottom**: #b38b3b 1px solid<br>• **font-family**: Arial<br>• **font-size**: 9pt<br>• **font-weight**: bold |

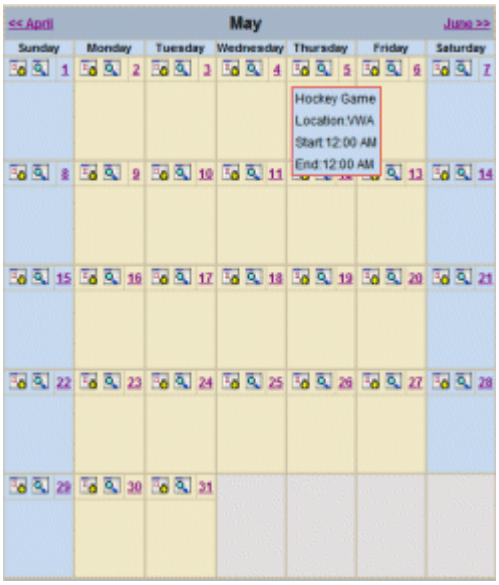
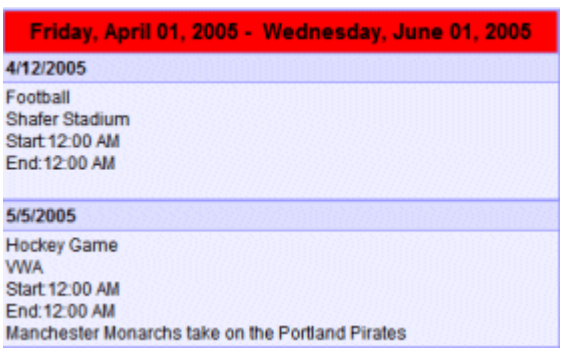
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_DateCellEmptyWeekend<br><br>**Note**: These cells appear empty when you are not logged in to Ektron CMS400.NET. | The weekend cells that hold the Add Event and View Event buttons. The following example shows the date cell empty weekend in red.<br> | • **background**: #C5DAEF<br>• **border-bottom**: #c9bda5 1px solid<br>• **font-family**: Arial<br>• **font-size**: 9pt<br>• **font-weight**: bold |
| .mv_DateCellEmptyToday<br><br>**Note**: This cell appears empty when you are not logged in to Ektron CMS400.NET. | The today cell that holds the Add Event and View Event buttons. The following example shows the date cell empty today in red.<br> | • **background**: #FFE354<br>• **border-bottom**: #B38B3B 1px solid<br>• **font-family**: Arial<br>• **font-size**: 9pt<br>• **font-weight**: bold |

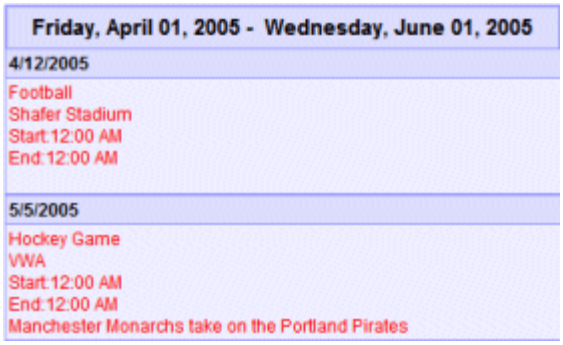
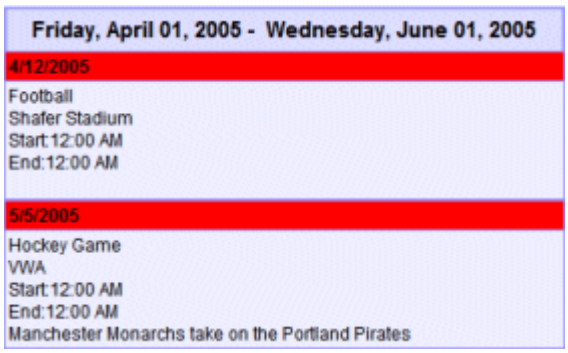
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_DateCellEmpty<br><br>**Note**: These cells appear empty when you are not logged in to Ektron CMS400.NET | The weekday cells that holds the Add Event and View Event buttons. The following example shows the date cell empty in red.<br>  | • **background**: #f0e9c5<br>• **border-bottom**: #c9bda5 1px solid<br>• **font-family**: Arial<br>• **font-size**: 9pt<br>• **font-weight**: bold |
| .mv_eventContainer | The popup window that appears when the event is hovered over. The following example shows the event container circled in red.<br> | • **visibility**: hidden<br>• **position**: absolute<br><br>Note: When the visibility property is set to visible, the event container appears upon page load. |

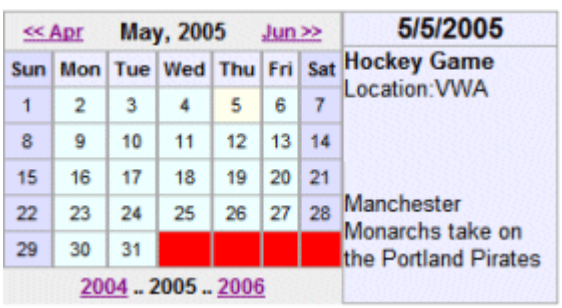
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_dayBorderToday | The today cell that holds listed events. The following example shows the day border today in red.<br> | • **border-right**: #b38b3b 2px solid<br>• **border-top**: #b38b3b 2px solid<br>• **background**: #ffe354<br>• **border-left**: #b38b3b 2px solid<br>• **border-bottom**: #b38b3b 2px solid |
| .mv_dayBorder<br>This class effects all weekday cells, except the today cell. | The weekday cell that holds listed events. The following example shows the day border in red.<br> | • **border-right**: #c9bda5 1px solid<br>• **border-top**: #c9bda5 1px solid<br>• **background**: #f0e9c5<br>• **border-left**: #c9bda5 1px solid<br>• **border-bottom**: #c9bda5 1px solid |

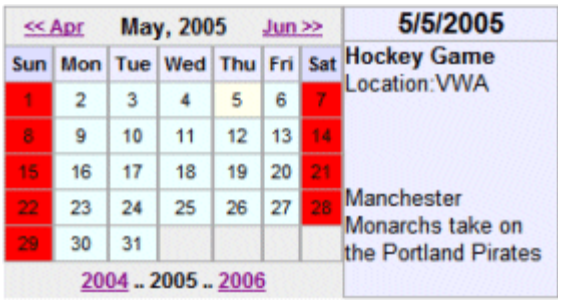
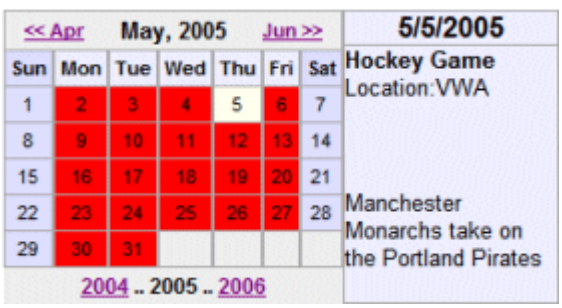
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_dayBorderWeekend | The Weekend day cell that holds listed events. The following example shows the day border weekend in red.<br> | • **border-right**: #c9bda5 1px solid<br>• **border-top**: #c9bda5 1px solid<br>• **background**: c5daef<br>• **border-left**: #c9bda5 1px solid<br>• **border-bottom**: #c9bda5 1px solid |
| .mv_eventTitle | The event title cell. The following example shows the event title cell in red.<br>  | • **font-weight**: normal<br>• **font-size**: 9pt<br>• **background**: transparent<br>• **font-family**: Arial |

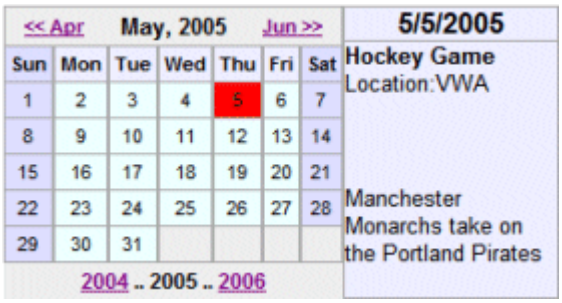
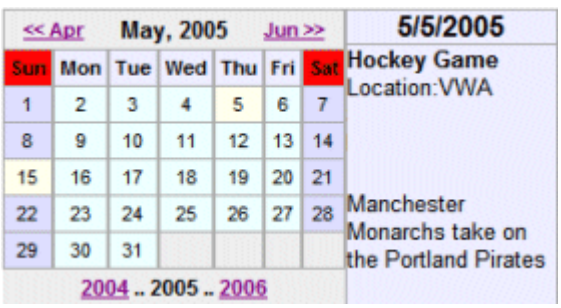
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_DayCellToday | Today's event title cell. The following example shows the day cell today in red.<br> | • **background**: transparent<br>• **font-family**: Arial<br>• **font-weight**: bold<br>• **font-size**: 9pt |
| .mv_TodayCell | The font for the event pop-up. The following example shows the today cell font in red.<br> | • **font-weight**: normal<br>• **font-size**: 9pt<br>• **background**: #C5DAEF<br>• **font-family**: Arial |

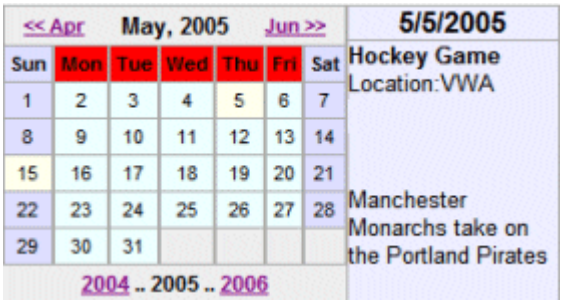
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mv_eventPopup | The border for the event pop-up. The following example shows the event pop-up border in red.<br>  | • **border-right**: #000000 1px solid<br>• **border-top**: #000000 1px solid<br>• **border-left** #000000 1px solid<br>• **border-bottom**: #000000 1px solid |
| ***Inline View (inline)*** | | |
| .ILViewHeaderBkg | The inline view header background. The following example shows the view header background in red.<br>  | • **border-right**: #9999ff 2px solid<br>• **border-top**: #9999ff 2px solid<br>• **font-weight**: bold<br>• **font-size**: 11pt<br>• **background**: #ddddff<br>• **border-left**: #9999ff 2px solid<br>• **font-family**: Arial<br>• **text-align**: center |

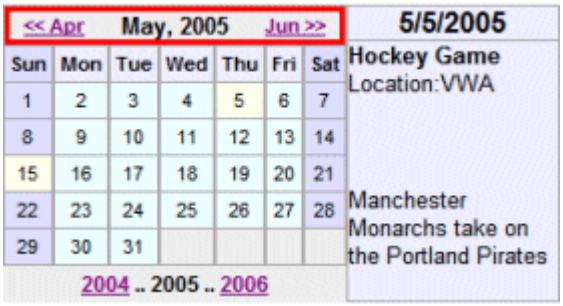
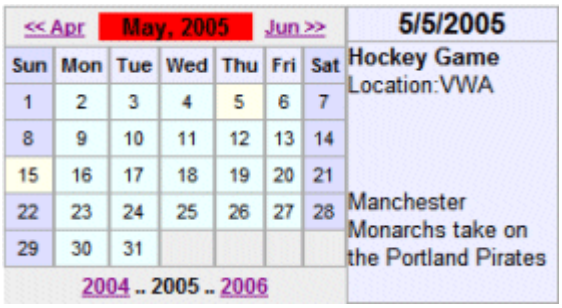
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .ILViewEvent | Font for the event description.The following example shows the view event font in red.<br>  | • **font-size**: 9pt<br>• **font-family**: Arial |
| .ILViewDate | The date for the inline view. The following example shows the view date background in red.<br>  | • **border-right**: #9999ff 1px solid<br>• **border-top**: #9999ff 1px solid<br>• **font-weight**: bold<br>• **font-size**: 9pt<br>• **background**: #ddddff<br>• **border-left**: #9999ff 1px solid<br>• **font-family**: Arial |

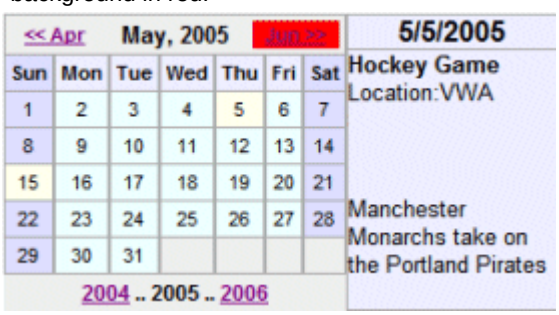
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .ILViewDayCell | The cell the events are listed in. The following example shows the view day cell in red.<br> | • **border-right**: #9999ff 1px solid<br>• **border-top**: #9999ff 1px solid<br>• **font-weight**: bold<br>• **font-size**: 9pt<br>• **background**: #eeeeff<br>• **border-left**: #9999ff 1px solid<br>• **border-bottom**: #9999ff 1px solid<br>• **font-family**: Arial |
| ***Month-Up with Event (monthupwithevent)*** | | |
| .mup_nonMonthCell | The cells at the beginning or end of the month that do not contain days. The following example shows the non-month cells in red.<br>  | • **background:** #eeeeee<br>• **font-family**: Arial<br>• **font-weight:** normal<br>• **font-size**: 8pt<br>• **text-align**: center<br>• **border-right**: #aaaaaa 1px solid<br>• **border-bottom**: #aaaaaa 1px solid<br>• **border-left**: #aaaaaa 1px solid |

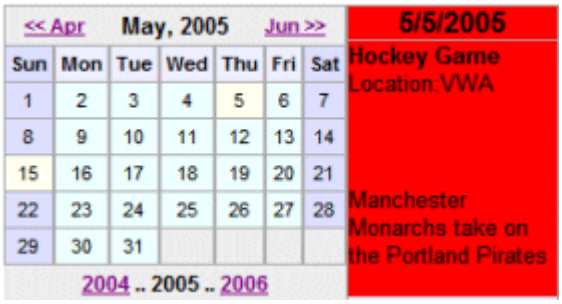
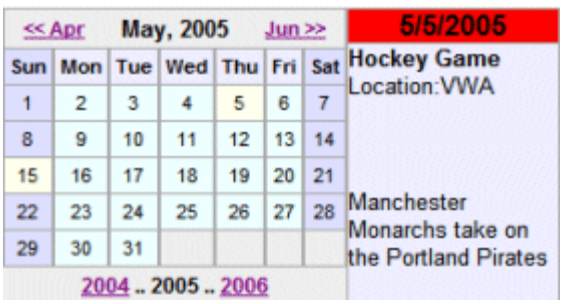
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mup_weekendCell | The weekend day cell. The following example shows the weekend cell in red.<br>  | • **background:** #dedeff<br>• **font-family**: Arial<br>• **font-weight:** normal<br>• **font-size**: 8pt<br>• **text-align**: center<br>• **border-right**: #aaaaaa 1px solid<br>• **border-bottom**: #aaaaaa 1px solid<br>• **border-left**: #aaaaaa 1px solid |
| .mup_weekdayCell | The week day cell. The following example shows the week day cell in red.<br>  | • **background:** #eeffff<br>• **font-family**: Arial<br>• **font-weight:** normal<br>• **font-size**: 8pt<br>• **text-align**: center<br>• **border-right**: #aaaaaa 1px solid<br>• **border-bottom**: #aaaaaa 1px solid<br>• **border-left**: #aaaaaa 1px solid |

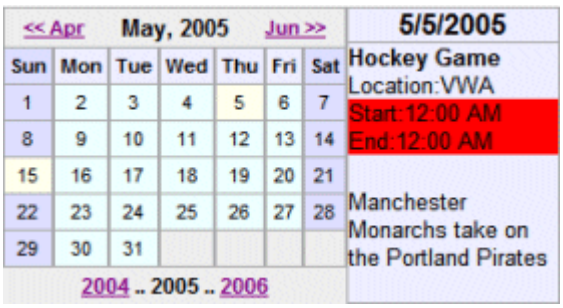
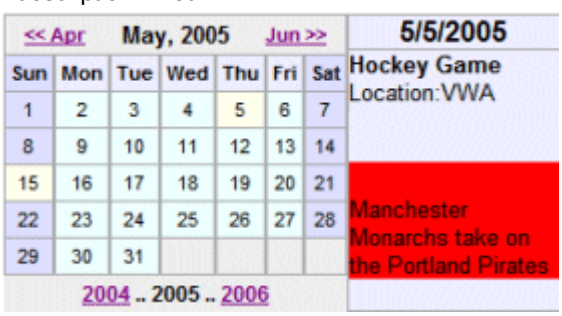
| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mup_weekdayEventCell | Highlights the weekdays that have events. The following example shows the weekday event cell in red.<br>  | • **background:** #ffffee<br>• **font-family**: Arial<br>• **font-weight:** normal<br>• **font-size**: 8pt<br>• **text-align**: center<br>• **border-right**: #aaaaaa 1px solid<br>• **border-bottom**: #aaaaaa 1px solid<br>• **border-left**: #aaaaaa 1px solid<br>• **cursor**: pointer |
| .mup_weekendDOWHeader | Weekend day of week header. The following example shows the weekend day of week header in red.<br>  | • **background:** #ddddff<br>• **font-family**: Arial<br>• **font-weight:** Bold<br>• **font-size**: 8pt<br>• **border-right**: #aaaaaa 1px solid<br>• **border-bottom**: #aaaaaa 1px solid<br>• **border-left**: #aaaaaa 1px solid<br>• **text-align**: center |

| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mup_weekdayDOWHeader | Weekday day of week header. The following example shows the weekday day of week header in red.<br>  | • **background:** #eeeeff<br>• **font-family**: Arial<br>• **font-weight:** Bold<br>• **font-size**: 8pt<br>• **border-right**: #aaaaaa 1px solid<br>• **border-bottom**: #aaaaaa 1px solid<br>• **border-left**: #aaaaaa 1px solid<br>• **text-align**: center |
| .mup_yearFooter | The year selection cell. The following example shows the year footer in red.<br> | • **background**: #eeeeee<br>• **font-family**: Arial<br>• **font-weight**: Bold<br>• **font-size**: 9pt<br>• **text-align**: center |

| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mup_monthHeaderBorder | The border around the month header. The following example shows the month header border in red.<br>  | • **background**: #eeeeee<br>• **font-family**: Arial<br>• **font-weight**: Bold<br>• **font-size**: 10pt<br>• **border-right**: #aaaaaa 1px solid<br>• **border-top**: #aaaaaa 1px solid<br>• **border-bottom**: #aaaaaa 1px solid<br>• **border-left**: #aaaaaa 1px solid<br>• **text-align**: center |
| .mup_monthHeader | The month header. The following example shows the month header in red.<br>  | • **background**: #eeeeee<br>• **font-family**: Arial<br>• **font-weight**: Bold<br>• **font-size**: 10pt<br>• **text-align**: center |

| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| ,mup_monthHeaderPrev | The previous button, on the month header. The following example shows the month header previous back ground in red.<br> | • **background**: #eeeeee<br>• **font-family**: Arial<br>• **font-weight**: Bold<br>• **font-size**: 8pt<br>• **text-align**: center |
| .mup_monthHeaderNext | The next button, on the month header. The following example shows the month header next background in red.<br>  | • **background**: #eeeeee<br>• **font-family**: Arial<br>• **font-weight**: Bold<br>• **font-size**: 8pt<br>• **text-align**: center |
| .mup_eventDispBkg<br><br>Note: .mup_eventDispBkg deals with the background layer of the event display. .mup_eventDisplay deals with the layer the event is displayed on. | The background for the event display. The following example shows the event display background in red.<br> | • **background**: #eeeeff<br>• **border-right**: #aaaaaa 1px solid<br>• **border-top**: #aaaaaa 1px solid<br>• **border-bottom**: #aaaaaa 1px solid<br>• **border-left**: #aaaaaa 1px solid |

| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mup_eventDisplay<br><br>Note: .mup_eventDispBkg deals with the background layer of the event display. .mup_eventDisplay deals with the layer the event is displayed on. | The event display. The following example shows the event display in red.<br>  | **background**: transparent |
| .mup_eventDisplayDate | The event display date header. The following example shows the event display date in red.<br>  | • **font-size**: 12pt<br>• **font-weight**: bold<br>• **background**: transparent<br>• **font-family**: Arial<br>• **text-align**: center<br>• **border-bottom**: #aaaaaa 1px solid |
| .mup_eventDisplayEventTitle | Display for the event title. The following example shows the event display event title in red.<br> | • **font-weight**: bold<br>• **background**: transparent<br>• **font-family**: Arial<br>• **font-size**: 10pt<br>• **text-align**: left |

| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .mup_eventDisplayLocation | Display for the event location. The following example shows the event display location in red.<br> | • **background**: transparent<br>• **font-family**: Arial<br>• **font-weight**: normal<br>• **font-size**: 10pt<br>• **text-align**: left |
| .mup_eventDisplayTimes | Display for event times. The following example shows the event display times in red.<br>  | • **background**: transparent<br>• **font-family**: Arial<br>• **font-weight**: normal<br>• **font-size**: 10pt<br>• **text-align**: left |
| .mup_eventDisplayLongDesc | Displays the event's long description. The following example shows the event display long description in red.<br>  | • **background**: transparent<br>• **font-family**: Arial<br>• **font-weight**: normal<br>• **font-size**: 10pt<br>• **text-align**: left |
| ***Event Type Control (used in month display)*** | | |

| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .etCtrl_Background | The background of the event type control cell. The following example shows the event type control background in red.<br> | • **border-right**: #c9bda5 2px solid<br>• **border-top**: #c9bda5 2px solid<br>• **border-left**: #c9bda5 2px solid<br>• **border-bottom**: #c9bda5 2px solid<br>• **font-size**: 9pt<br>• **background**: #A5B7C9<br>• **font-family**: Arial |
| .etCtrl_InstructCell | The instructions cell for event type control. The following example shows the instruction cell in red.<br> | • **background**: transparent<br>• **font-family**: Arial<br>• **font-size**: 9pt<br>• **font-weight**: bold<br>• **text-align**: right<br>• **vertical-align**: top |

| Style Class | Description | Attributes and Default Settings |
|---|---|---|
| .etCtrl_SelCell | The cell the selection box is displayed in. The instructions cell for event type control. The following example shows the selection cell in red.<br> | • **background**: transparent<br>• **font-family**: Arial<br>• **font-size**: 9pt<br>• **text-align**: left<br>• **vertical-align**: top |
| .etCtrl_selectBox | The event type selection box. The instructions cell for event type control. The following example shows the selection box in red.<br> | • **background**: #c5daef |

# Menu Server Control

The Menu Server Control calls a menu to be displayed on a page. Using the Menu Server Control, you can manipulate a menu by using the DisplayXslt property. Below is a menu display with the SampleMenu Xslt.

Before you can use the Menu Server Control, you must create a menu in the **Ektron CMS400.NET** work area. To learn about creating menus, see “Working with Menus” in the **Ektron CMS400.NET** User Manual.

This subsection contains the following topics:

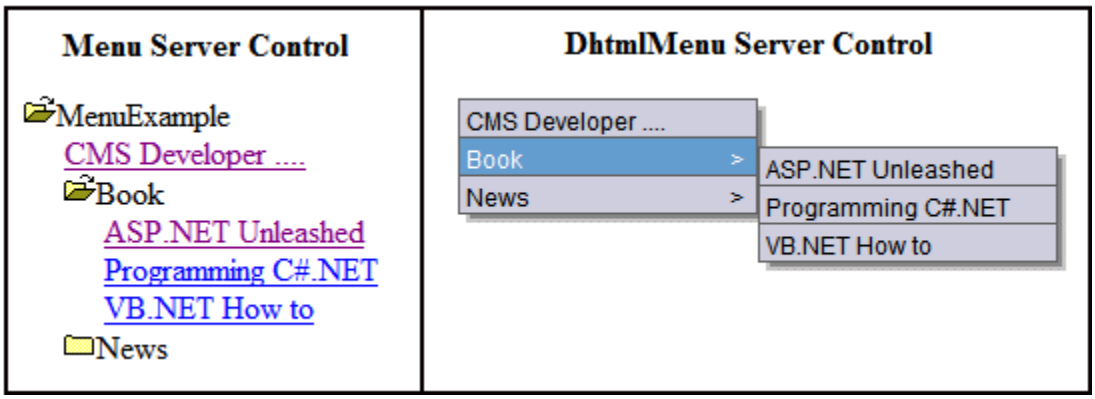

- ”Menu Server Control vs. DhtmlMenu Server Control” on page 161
- ”Menu Server Control Properties” on page 162
- ”Using DisplayXslt Samples” on page 164


## Menu Server Control vs. DhtmlMenu Server Control

The Menu Server Control calls a menu to be displayed on a page. You can manipulate the menu by using the DisplayXslt property.

With the DhtmlMenu Server Control, you display a dynamic menu with drop down boxes and submenus. You manipulate DhtmlMenus with a style sheet. DhtmlMenus save space on the Web page by having the drop down menus and submenus float over content.

**WARNING!** You cannot create a DHTML menu using the code behind -- you must drag and drop it. Once the menu is on the template, you can manipulate its properties using the code behind.




# Menu Server Control Properties

The Menu Server Control properties are described in this table.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc.<br>*See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | String |

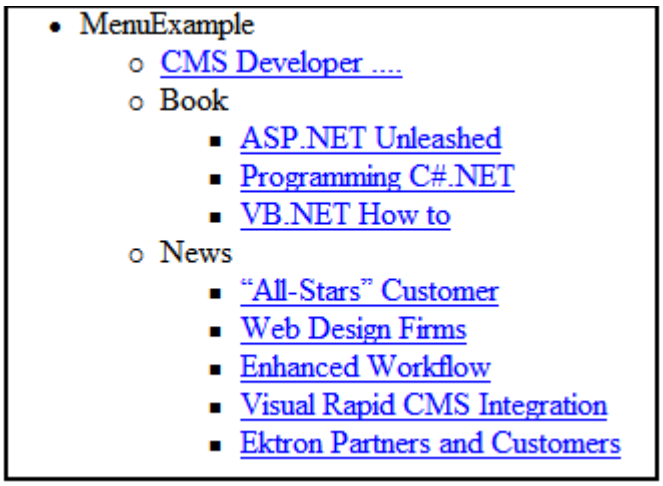
| Property | Value | Data Type |
|---|---|---|
| DefaultMenuID | The ID of a menu that appears where you insert this server control if no other menu is identified or available.<br>If you don't know the ID number of the menu, use the CMS Explorer to browse to it.<br>*See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | Integer |
| DisplayXslt | The Xslt to use to render the menu.<br>**None** - databind only<br>SampleMenu - A sample display, formatted as a bulleted menu list<br>TreeMenu - A sample display, formatted as a folder tree. You can expand the tree by clicking on the folder icon.<br>**Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page<br>For more information on using the SampleMenu and TreeMenu DisplayXslt, see "Using DisplayXslt Samples" on page 164. | String |
| DynamicParameter | Select **menu_id**. When you do, this server control uses the menu passed as a URL parameter. | String |
| Hide | Used to hide a menu in design time and run time.<br>**True** = Hide menu<br>**False** = Show menu | Boolean |
| Language | Design time only: View content blocks, collections, etc. in different languages<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an inline portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |

# Using DisplayXslt Samples

Two DisplayXslt samples are provided with the Menu Server Control, SampleMenu and TreeMenu. This section explains how to use them.

## SampleMenu

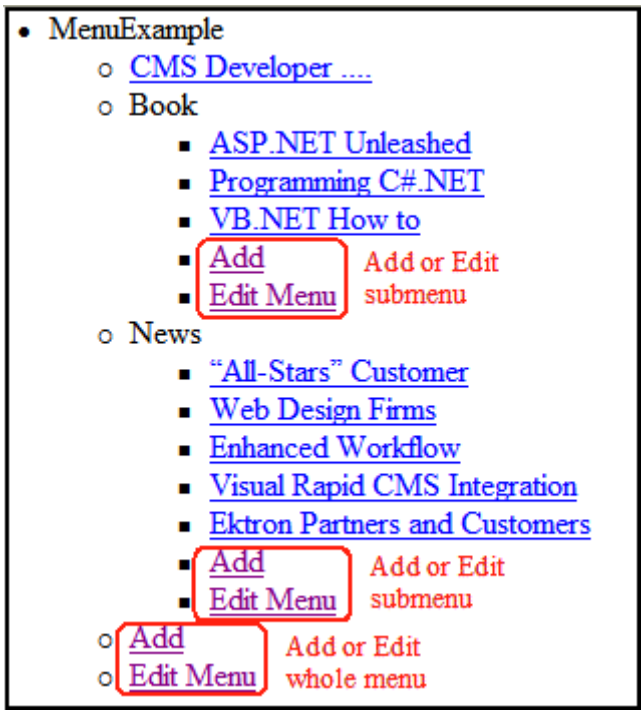
The SampleMenu DisplayXslt lets you display an **Ektron CMS400.NET** menu as a bulleted item list. See below for an example.




When you are logged in to your **Ektron CMS400.NET** site, two menu items are added to each section of the menu: Add and Edit Menu. The user can use these to add a new menu item or edit an existing one. See example below.

For Information on a adding a menu Item, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Adding a New Content Block to a Menu via Navigation Link on a Web Page".

For Information on a editing a menu, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Editing a Menu via Navigation Link on a Web Page".




### *SampleMenu XSL code*

Below is the XSL code that is used to create SampleMenu Display. You can use this code as the basis to design your own XSLT.

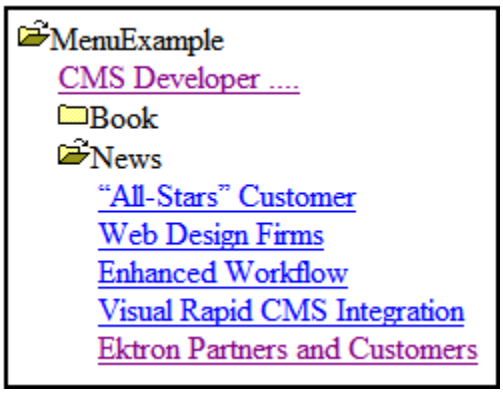
```
<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/
XSL/Transform">

 <xsl:output method="xml" version="1.0" encoding="UTF-8"
indent="yes" omit-xml-declaration="yes"/>

 <xsl:template match="/">

  <xsl:apply-templates select="MenuDataResult/Item"/>

 </xsl:template>

  <xsl:template match="Item[parent::MenuDataResult]">

  <xsl:call-template name="makelist"/>

 </xsl:template>

 <xsl:template match="Item[child::Menu]">
```

```
  <LI>

  <xsl:apply-templates select="Menu"/>

  </LI>

 </xsl:template>

 <xsl:template name="makelist" match="Menu">

  <xsl:choose>

  <xsl:whentest="Link=''"><xsl:value-of select="Title"/></
xsl:when><xsl:otherwise>

  <A><xsl:attribute name="href"><xsl:value-of
select="concat('',Link)"/></xsl:attribute>

  <xsl:value-of select="Title"/></A></xsl:otherwise></xsl:choose>

  <UL><xsl:apply-templates select="Item"/></UL>

 </xsl:template>

 <xsl:template match="Item">

  <xsl:param name="pItemLink"><xsl:value-of select="ItemLink"/></
xsl:param>

  <xsl:choose><xsl:whentest="ItemLink=''">

  <LI>

  <xsl:value-of select="ItemTitle"/>

  <xsl:apply-templates select="Item|Menu"/></LI>

  </xsl:when>

  <xsl:otherwise>

  <LI><A>

  <xsl:choose>

   <xsl:whentest="ItemType='ExternalLink'">

      <xsl:attribute name="href"><xsl:value-of select="ItemLink"/></
xsl:attribute>

    <xsl:value-of select="ItemTitle"/>

   </xsl:when>

   <xsl:whentest="ItemType='Javascript'">
```

```
    <xsl:attribute name="onClick">Javascript:<xsl:value-of
select="ItemLink"/></xsl:attribute>

    <xsl:attribute name="href">#</xsl:attribute>

    <xsl:value-of select="ItemTitle"/>

   </xsl:when>

   <xsl:otherwise>

    <xsl:attribute name="href">

    <xsl:value-of select="concat('',ItemLink)"/>

    </xsl:attribute>

    <xsl:value-of select="ItemTitle"/>

   </xsl:otherwise>

  </xsl:choose></A><xsl:apply-templates select="Item|Menu"/></LI></
xsl:otherwise></xsl:choose>

 </xsl:template>

 <xsl:template name="maketitle"><xsl:choose>

  <xsl:whentest="ItemLink=''"><xsl:value-of select="Title"/></
xsl:when><xsl:otherwise><A>

  <xsl:attribute name="href"><xsl:value-of
select="concat('',Link)"/></xsl:attribute>

  <xsl:value-of select="Title"/></A></xsl:otherwise></xsl:choose>

  <UL><xsl:apply-templates select="Item"/></UL>

 </xsl:template>

 <!-- identity template -->

 <xsl:template match="@*|node()">

  <xsl:copy><xsl:apply-templates select="@*|node()"/></xsl:copy>

 </xsl:template>

</xsl:stylesheet>
```

## TreeMenu

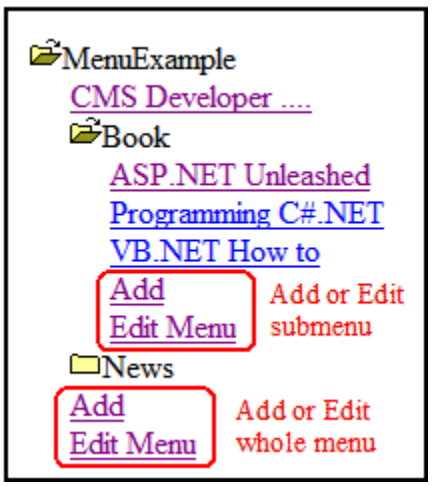
The TreeMenu DisplayXslt lets you display an **Ektron CMS400.NET** menu as a clickable folder list. You can expand and collapse the menu by clicking on the folder icons. See below for an example.




When you are logged in to your **Ektron CMS400.NET** site, two menu items are added to each section of the menu: Add and Edit Menu. The user can use these to add a new menu item, or edit an existing menu item. See example below.

For Information on a adding a menu Item, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Adding a New Content Block to a Menu via Navigation Link on a Web Page".

For Information on a editing a menu, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Editing a Menu via Navigation Link on a Web Page".



**NOTE** When using the TreeMenu Server Control with Visual Studio and Windows 2003, the folder images do not display during design-time. They do, however, display correctly during run-time.

### *TreeMenu XSL code*

Below is the XSL code that is used to create TreeMenu Display. You can use this code as the basis to design your own XSLT.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

    <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/
1999/XSL/Transform">

    <xsl:output method= "xml" version="1.0" encoding="UTF-8"
indent="yes" omit-xml-declaration= "yes"/>

<!-- Change sitepath to match you environment -->

<xsl:variable name="sitepath">/CMS400Example/</xsl:variable>

<!--

menuId must be different for each menu on the page.

This does mean that you need to use a different copy

of this file (each with their own menuId) for each

menu that you will display.
```

```
-->

<xsl:variable name="menuId">Menu1</xsl:variable>

<xsl:template match="/">

<xsl:apply-templates select="MenuDataResult/Item"/>

</xsl:template>

<xsl:template match="Item[parent::MenuDataResult]">

<xsl:apply-templates select="Item/Menu"/>

</xsl:template>

<xsl:template match="Item[child::Menu]">

<xsl:apply-templates select="Menu"/>

</xsl:template>

<xsl:template match="Menu">

<xsl:choose>

<xsl:whentest="Link=''">

<span>

<xsl:attribute name="class">handcur</xsl:attribute>

<xsl:attribute name="onclick">showBranch('<xsl:value-of
select="$menuId" />_<xsl:value-of select="ID"/>')</xsl:attribute>

<img>

<xsl:attribute name="src"><xsl:value-of select="$sitepath" />
>workarea/images/application/ico_menu-closed.gif</xsl:attribute>

<xsl:attribute name="id">I<xsl:value-of select="$menuId" />
>_<xsl:value-of select="ID"/></xsl:attribute>

</img>

<xsl:choose>

<xsl:whentest="ImageOverride='True'">

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="Image"/></
xsl:attribute>

</img>
```

```
</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(Image='')">

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="Image"/></
xsl:attribute>

</img><xsl:text> </xsl:text>

</xsl:if>

<xsl:value-of select="Title"/>

</xsl:otherwise>

</xsl:choose>

</span>

<br/>

</xsl:when>

<xsl:when test="not(Link)">

</xsl:when>

<xsl:otherwise>

<span>

<xsl:attribute name="class">handcur</xsl:attribute>

<xsl:attribute name="onclick">showBranch('<xsl:value-of
select="$menuId" />_<xsl:value-of select="ID"/>')</xsl:attribute>

<img>

<xsl:attribute name="src"><xsl:value-of select="$sitepath" />
>Workarea/images/application/ico_menu-closed.gif</xsl:attribute>

<xsl:attribute name="id">I<xsl:value-of select="$menuId" />
>_<xsl:value-of select="ID"/></xsl:attribute>

</img>

<xsl:choose>

<xsl:whentest="ImageOverride='True'">
```

```
<A><xsl:attribute name="href"><xsl:value-of
select="concat('',Link)"/></xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="Image"/></
xsl:attribute>

</img>

</A>

</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(Image='')">

<A><xsl:attribute name="href"><xsl:value-of
select="concat('',Link)"/></xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="Image"/></
xsl:attribute>

</img>

</A><xsl:text> </xsl:text>

</xsl:if>

<A><xsl:attribute name="href"><xsl:value-of
select="concat('',Link)"/></xsl:attribute>

<xsl:value-of select="Title"/>

</A>

</xsl:otherwise>

</xsl:choose>

</span>

<br/>

</xsl:otherwise>

</xsl:choose>

<span class="branch">
```

```
<xsl:attribute name="id"><xsl:value-of select="$menuId" />
>_<xsl:value-of select="ID"/></xsl:attribute>

<xsl:apply-templates select="Item"/>

</span>

</xsl:template>

<xsl:template match="Item">

<xsl:choose>

<xsl:whentest="ItemLink=''">

<span class="handcur">

<xsl:choose>

<xsl:whentest="ItemImageOverride='True'">

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(ItemImage='')">

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img><xsl:text> </xsl:text>

</xsl:if>

<xsl:value-of select="ItemTitle"/>

</xsl:otherwise>

</xsl:choose>

</span><br/>

<xsl:apply-templates select="Item|Menu"/>
```

```
</xsl:when>

<xsl:otherwise>

<span>

<xsl:attribute name="class">handcur</xsl:attribute>

<xsl:choose>

<xsl:whentest="ItemType='ExternalLink'">

<xsl:choose>

<xsl:whentest="ItemImageOverride='True'">

<A><xsl:attribute name="href"><xsl:value-of select="ItemLink"/></
xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A>

</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(ItemImage='')">

<A><xsl:attribute name="href"><xsl:value-of select="ItemLink"/></
xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A><xsl:text> </xsl:text>

</xsl:if>

<A><xsl:attribute name="href"><xsl:value-of select="ItemLink"/></
xsl:attribute>

<xsl:value-of select="ItemTitle"/>
```

```
</A>

</xsl:otherwise>

</xsl:choose>

</xsl:when>

<xsl:whentest="ItemType='Javascript'">

<xsl:choose>

<xsl:whentest="ItemImageOverride='True'">

<A><xsl:attribute name="onClick">Javascript:<xsl:value-of
select="ItemLink"/></xsl:attribute><xsl:attribute name="href">#</
xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A>

</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(ItemImage='')">

<A><xsl:attribute name="onClick">Javascript:<xsl:value-of
select="ItemLink"/></xsl:attribute><xsl:attribute name="href">#</
xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A><xsl:text> </xsl:text>

</xsl:if>

<A><xsl:attribute name="onClick">Javascript:<xsl:value-of
select="ItemLink"/></xsl:attribute><xsl:attribute name="href">#</
xsl:attribute>

<xsl:value-of select="ItemTitle"/>
```

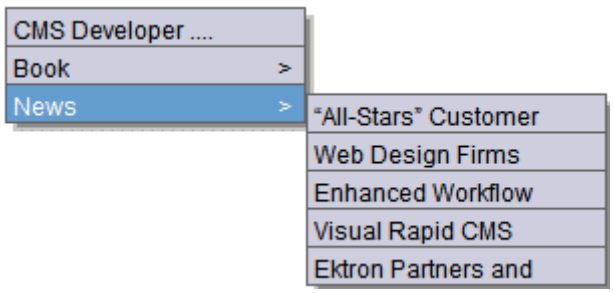
```
</A>

</xsl:otherwise>

</xsl:choose>

</xsl:when>

<xsl:otherwise>

<xsl:choose>

<xsl:whentest="ItemImageOverride='True'">

<A><xsl:attribute name="href"><xsl:value-of
select="concat('',ItemLink)"/></xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A>

</xsl:when>

<xsl:otherwise>

<xsl:iftest="not(ItemImage='')">

<A><xsl:attribute name="href"><xsl:value-of
select="concat('',ItemLink)"/></xsl:attribute>

<img>

<xsl:attribute name="border">0</xsl:attribute>

<xsl:attribute name="src"><xsl:value-of select="ItemImage"/></
xsl:attribute>

</img>

</A><xsl:text> </xsl:text>

</xsl:if>

<A><xsl:attribute name="href"><xsl:value-of
select="concat('',ItemLink)"/></xsl:attribute>

<xsl:value-of select="ItemTitle"/>

</A>
```

```
</xsl:otherwise>

</xsl:choose>

</xsl:otherwise>

</xsl:choose></span><br/>

<xsl:apply-templates select="Item|Menu"/>

</xsl:otherwise>

</xsl:choose>

</xsl:template>

<!-- identity template -->

<xsl:template match="@*|node()">

<xsl:copy><xsl:apply-templates select="@*|node()"/></xsl:copy>

</xsl:template>

</xsl:stylesheet>
```

# DhtmlMenu Server Control

**NOTE** For optimal performance, Ektron recommends using at minimum Internet Explorer 5.5, Netscape 6.0, or Mozilla Firefox 1.0.

The DhtmlMenu Server Control lets you display a menu with drop down boxes and submenus. This lets developers add a menu to a page without taking up a lot space. You can modify the menu's style via a style sheet.

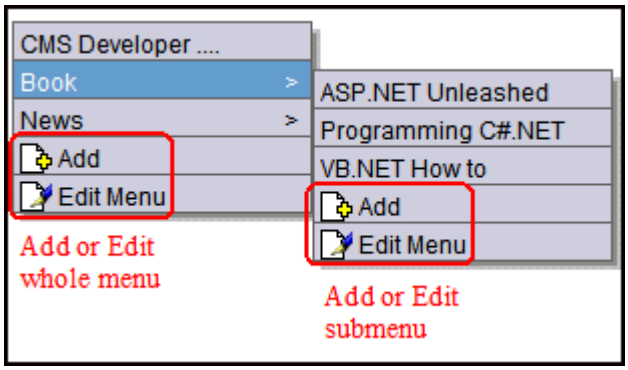



Before using the DhtmlMenu Server Control, you must add items to a menu in the Ektron CMS400.NET workarea. To learn about creating menus, see "Working with Menus" in the CMS400.NET User Manual.

The DhtmlMenu Server Control also lets developers add content to a menu. When you are logged in to your **Ektron CMS400.NET** site, two menu items are added to each menu: Add and Edit Menu. The user can use these to add a new menu item or edit an existing menu item. See example below.

For Information on a adding a menu Item, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Adding a New Content Block to a Menu via Navigation Link on a Web Page".

For Information on a editing a menu, see **Ektron CMS400.NET** Users Manual > "Working with Menus" > "Managing Menus" > "Editing a Menu via Navigation Link on a Web Page".




This subsection contains the following topics:


- "Menu Server Control vs. DhtmlMenu Server Control" on page 179
- "DhtmlMenu Server Control Properties" on page 179
- "Using a Style Sheet" on page 183

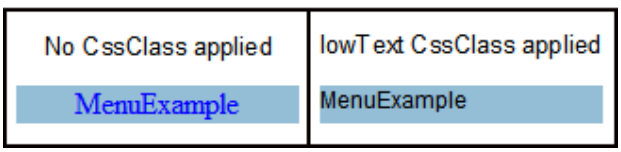

# Menu Server Control vs. DhtmlMenu Server Control

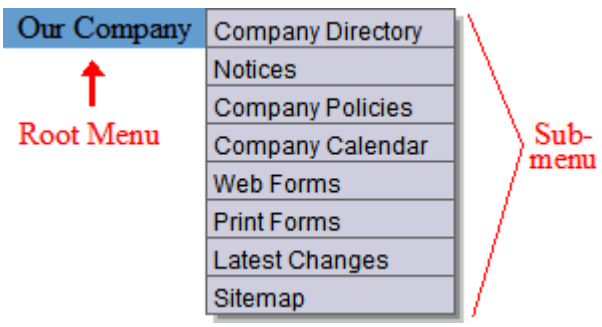
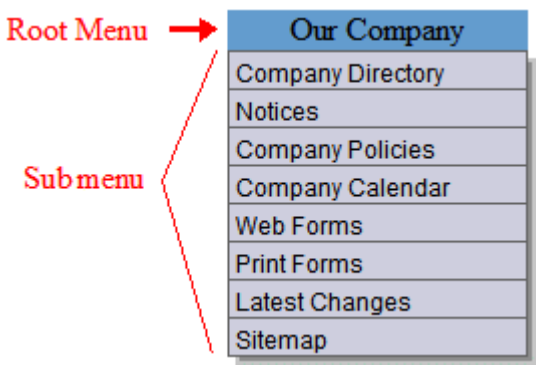
See "Menu Server Control vs. DhtmlMenu Server Control" on page 161.

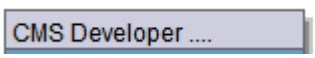
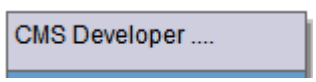
# DhtmlMenu Server Control Properties

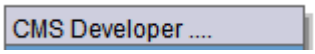
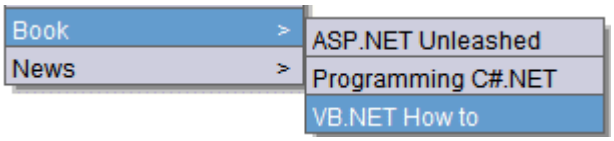
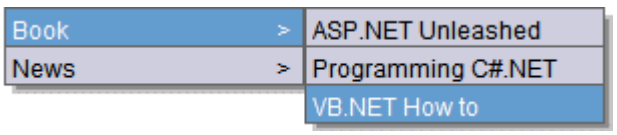
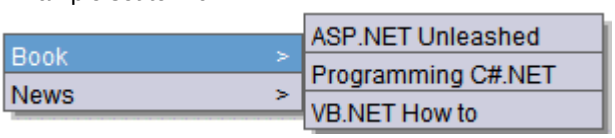
The DhtmlMenu Server Control properties are described in this table.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| Align | Specifies the horizontal alignment of the root menu text.<br>The default is **Center.**<br>**NotSet** - Use the default setting.<br>**Left** - Aligns text to the left.<br>**Center** - Aligns text to the center.<br>**Right** - Aligns text to the right.<br>**Justify** - Aligns text justified to both margins. | String |
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc.... *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | String |
| BackColor | Sets the background color for the root menu. This property appears in the **Appearance** category. | String |
| CssClass | Apply a .css class to the root menu. Enter the class name from the .css defined in the Stylesheet property. *See Also:* "Stylesheet" on page 182<br>Since pop_style.css is the default style sheet, you can use it's classes without defining it in the Stylesheet property.<br>For example enter lowtext in the property, the root menu changes as shown below.<br>  | String |
| DefaultMenuID | The ID of a menu that appears where you inserted this server control if no other content block is identified, or is not available.<br>If you don't know the ID number of the menu, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | Integer |
| DynamicParameter | Select **menu_id**. When you do, this server control uses the menu passed as a URL parameter. | String |

| Property | Value | Data Type |
|---|---|---|
| ForeColor | Sets the color of the font for the root menu. This property appears in the **Appearance** category. | |
| Hide | Used to hide a menu in design time and run time.<br>**True** = Hide menu<br>**False** = Show menu | Boolean |
| HorizontalDisplay | Displays submenu to the right of the root menu. This only affects the first submenu of the root menu. All subsequent submenus appear to the right of each submenu.<br>The default is **False**<br>**True** = Submenu appears to the right of the root menu<br>  <br><br>**False** = Submenu appears below the root menu<br>  <br> | Boolean |

| Property | Value | Data Type |
|---|---|---|
| Language | Design time only: View content blocks, collections, etc. in different languages<br><br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |
| MenuHoverOverColor | Sets the background color for the root menu when hovering over it. | String |
| MenuStyle | A specified inline style that defines the root menu.<br>For example enter the following in the MenuStyle property box:<br>`font: bold 18px Arial;color:royalblue; background-color:slateblue`<br> | String |
| Stylesheet | Specify the location and style sheet for the submenu. For example: `\Workarea\csslib\mytest.css.`<br>Leave blank to use the default.<br>`pop_style.css` is the default style sheet. It is located in `\webroot\CMS400Example\Workarea\csslib`. For more information on style sheets, see "Using a Style Sheet" on page 183<br>The Stylesheet property can be used in conjunction with the CssClass property to define the root menu. *See Also:* "CssClass" on page 180 | String |
| SubMenuBackColor | Sets the background color for the submenu. | String |
| SubMenuHoverOverColor | Sets the background color for the submenu when hovering over it. | String |
| SubMenuItemHeight | Sets the height of each submenu item in pixels. The Default is 20<br><br>Example set to 20:  <br><br>Example set to 30:  | Integer |

| Property | Value | Data Type |
|---|---|---|
| SubMenuItemWidth | Sets the width of each submenu item in pixels. The Default is 150<br>Example set to 150:  <br>Example set to 130: | Integer |
| SubMenuTopOffset | Sets the location of the top of the submenu. This number is in pixels. The default is 5<br>Example set to 5:  <br>Example set to 0 (zero):  <br>Example set to -10:  | Integer |
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an inline portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |

## Using a Style Sheet

You can create your own style sheet, or modify the existing default one, `pop_style.css`. It is located in `\webroot\CMS400Example\Workarea\csslib`.

Style Sheets let you choose different text colors, text alignment, text fonts, and text size to create a customized menu.

> Best Practice
>
> Ektron recommends copying and renaming the default style sheet. This insures you always have a clean file to start with. This also gives you something to reference if you are not getting the results you expect.

The style sheet must be located in the site root.

You can only define one menu style sheet for each Web page. If you have more than one menu on a single Web page, they share the style sheet.

The following is a list of style sheet properties that can be modified in the `pop_style.css`.

| Style Class | Description | Attribute | Default Value |
|---|---|---|---|
| .lowText | Text of the floating menu as displayed on the menu. | • color<br>• font<br>• text-align<br>• text-decoration | • #000000 (Black)<br>• 12px Arial, sans-serif<br>• left<br>• none |
| .highText | Text of the floating menu when highlighted. | • color<br>• font<br>• text-align<br>• text-decoration | • #000000 (Black)<br>• 12px Arial, sans-serif<br>• left<br>• none |

| Style Class | Description | Attribute | Default Value |
|---|---|---|---|
| .roothighText<br><br>Note: This class only works when called by the CssClass property. Use the Font property in the Appearance category to define root menu text. | Text of the root menu as displayed on a Web page. | • color<br>• font<br>• text-align<br>• text-decoration | • #000000 (Black)<br>• 16px Arial, sans-serif<br>• center<br>• none |
| .itemBorder | Border color and size of the menu. | • border | • 1px solid #666666 |
| .itemBorderBlank | Separator border color and size of the menu. | • border | • 1px solid #CCCCDD |

For more information on style sheets, visit http://www.w3.org/Style/CSS/.

# MetaData Server Control

The Metadata Server Control lets you add the metadata from content blocks to a Web page. This lets developers add metadata quickly without having to type it in. You can add metadata from a single content block, multiple content blocks, or by dynamically passing a content ID from a URL.

This subsection contains the following topics:


- ”MetaData Server Control vs. MetaDataList Server Control” on page 186
- ”MetaData Server Control Properties” on page 186
- ”Using the MetaData Server Control” on page 187


## MetaData Server Control vs. MetaDataList Server Control

With the MetaData Server Control, you add metadata from content blocks to your Web page. With the MetaDataList Server Control, you create a list of content blocks to display on your site, based on the Metadata in each content block. For Information on the MetaDataList Server Control, see ”MetaDataList Server Control” on page 189.

## MetaData Server Control Properties

The Metadata Server Control properties are described in this table.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | String |
| DefaultContentID | The ID of the content block that the server control will get the metadata from.<br>If you want to add metadata from several content blocks, set this property to zero (0) and use the DefaultItemList property to identify the content blocks.<br>If you don't know the ID number of the content block, use the CMS Explorer to browse to it. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | Integer |
| DefaultItemList | A comma delimited list of content block IDs to get metadata from. DefaultContentID must be set to zero (0) so you can add IDs to the list.<br> | Integer |
| DynamicParameter | Gets or sets the QueryString parameter to read a content ID dynamically. | String |
| Hide | Used to hide a Metadata Server Control in design time and run time.<br>**True** = Hide Metadata Server Control<br>**False** = Show Metadata Server Control | boolean |

# Using the MetaData Server Control

These steps show how to use the MetaData Server Control.

1. Drag a MetaData Server Control into a template.
2. Set the properties of the Metadata Server Control. This will create the following HTML in the HTML body.

```
        <cms:MetaData id="MetaData1" runat="server"
DefaultContentID="12"></cms:MetaData>
```

Or if you are using multiple content block IDs In the DefaultItemList, the following HTML will be created.

```
        <cms:metadata id="MetaData1" runat="server"
DefaultItemList="[12,7,4]"></cms:metadata>
```

3. Click on the HTML tab and copy that line from the <body> tag into the <head> tag.

```
<HEAD>
    <title>WebForm5</title>
    <cms:MetaData id="Metadata2" runat="server" DefaultContentID="12"></cms:MetaData>
    <meta name="GENERATOR" content="Microsoft Visual Studio .NET 7.1">
    <meta name="CODE_LANGUAGE" content="Visual Basic .NET 7.1">
    <meta name="vs_defaultClientScript" content="JavaScript">
    <meta name="vs_targetSchema" content="http://schemas.microsoft.com/intellisense/i
</HEAD>
<body>
    <form id="Form1" method="post" runat="server">
    </form>
</body>
```

4. Save the Web form and rebuild the solution.
5. Open the Web page in the browser.
6. Right click on the Web page and click **View Source**. Look in the head tag. The meta tags from the content block are added to the page.

The metadata information added is shown below.

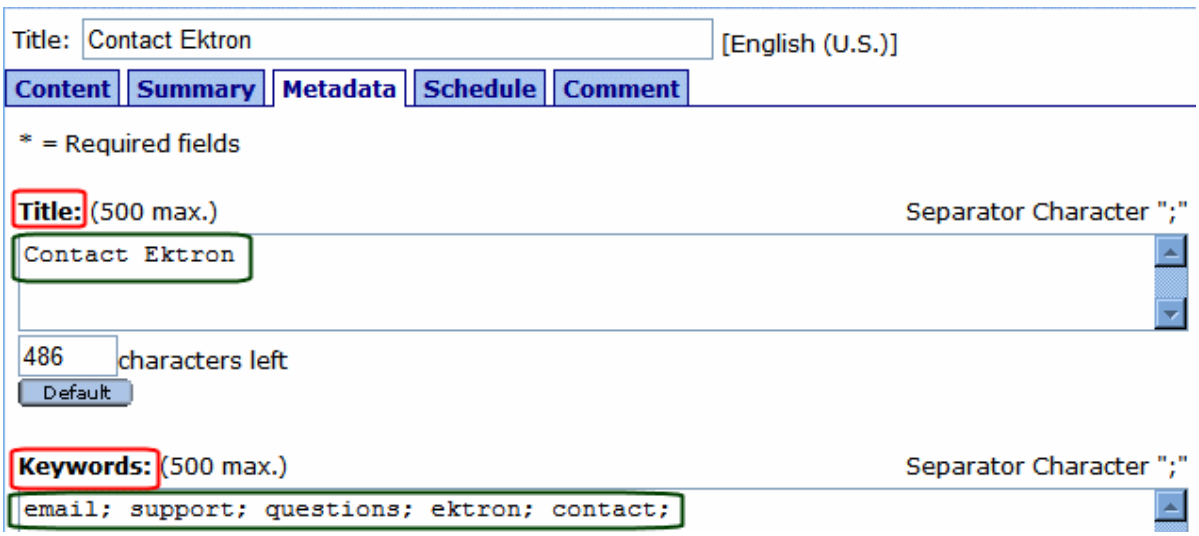
WebForm5[1] - Notepad

File Edit Format View Help

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
  <HEAD>
    <title>WebForm5</title>
    <title>CMS Developer</title>

    <meta name="keywords" content="CMS400; .NET; CMS">
    <meta name="description" content="CMS400 developer section">

    <meta content="Microsoft Visual Studio .NET 7.1" name="GENERATOR">
    <meta content="Visual Basic .NET 7.1" name="CODE_LANGUAGE">
    <meta content="JavaScript" name="vs_defaultClientScript">
    <meta content="http://schemas.microsoft.com/intellisense/ie5"
name="vs_targetSchema">
  </HEAD>
```

# MetaDataList Server Control

Use the MetaDataList Server Control to create lists based on Keyword Names and Keyword Values contained within the metadata of a content block. In the example below the KeyWordNames are circled in red and the KeyWordValues are circled in green.




The list can display the information as a list of hyperlinks. You can choose, based on properties you set, if you would like to have a summary displayed. You can also choose how to order the display. For information on adding Metadata to a content block, see the Users Manual > “Adding or Editing Metadata”.

This subsection contains the following topics:


- ”MetaData Server Control vs. MetaDataList Server Control” on page 190
- ”MetaDataList Server Control Properties” on page 190

# MetaData Server Control vs. MetaDataList Server Control

With the MetaData Server Control, you add metadata from content blocks to your Web page. With the MetaDataList Server Control, you create a list of content blocks to display on your site based on the Metadata in each content block. For Information on the MetaData Server Control, see ”MetaData Server Control” on page 186

# MetaDataList Server Control Properties

The MetaDataList Server Control properties are described in this table.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* ”Using CMS Explorer to Browse Your Ektron CMS400.NET Site” on page 32 | String |
| ContentType | One of the following<br>• all types<br>• archived content<br>• archived forms<br>• content<br>• forms<br>To learn about archived content, see the **Ektron CMS400.NET** User Manual, “Scheduling Content to Begin and End” > “Setting Archive Options.” | String |

| Property | Value | Data Type |
|---|---|---|
| DisplayXslt | Determines how the information on the page is displayed<br>**None**-databind only<br>**ecmNavigation** - lists the title of every content block in the folder<br>*See Also:* "Example of ecmNavigation Display" on page 122<br>**ecmTeaser** - lists the title of every content block in the folder plus the content summary.<br>*See Also:* "Example of ecmTeaser Display" on page 123<br>**Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page | String |
| ExactPhrase | Determines whether the KeyWordValue needs to match the phrase exactly.<br>For example, if "site" is the KeyWordValue and the title of a content block is "Welcome to the site" and ExactPhrase is set to true, you would not see the content block in the metadata list. This is because "site" does not equal "Welcome to the site".<br>**True** = Match the exact phrase<br>**False** = Doesn't need to match exact phrase | Boolean |
| FolderID | The root folder content will be pulled from. | Integer |
| GetHtml | Set to **True** if you want to retrieve and display content (html body) for all content blocks in the collection. For example, display content inside a web server control such as a DataGrid. | Boolean |
| Hide | Used to hide a metadata list in design time and run time.<br>**True** = Hide metadata list<br>**False** = Show metadata list | Boolean |

| Property | Value | Data Type |
|---|---|---|
| KeyWordName | The KeyWordName is the container where the KeyWordValues are located.<br>Some examples of a KeyWordName might be “Keywords or Title”<br>For information on creating KeyWordNames for Metadata, see the **Ektron CMS400.NET** Administrators Manual > “Managing Content” > “Metadata” > “Managing Metadata Definitions” > “Adding a Metadata Definitions” | String |
| KeyWordValue | The value that is associated with the KeyWordName<br>Some examples of a KeyWordValue might be “Home; page; or company” | String |
| Language | Design time only: View content blocks, collections, etc. in different languages.<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |
| MaxNumber | The maximum number of results returned for display.<br>**0** = unlimited | Integer |
| OrderBy | The order of the list to be returned.<br>• **Title** - The title of the content block<br>• **ID** - The content block ID number<br>• **Date Created** - The date the content block was created<br>• **Date Modified** - The date the content block was last modified<br>• **LastEditorLname** - The last editor’s last name<br>• **LastEditorFname** - The last editor’s first name | String |
| Recursive | Whether to search sub-folders of the identified root folder. The root folder is identified in the Folderpath property. | Boolean |
| SortOrder | Ascending or Descending | String |

| Property | Value | Data Type |
|---|---|---|
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an inline portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |

# RssAggregator Server Control

An RSS Aggregator or reader takes an RSS feed and displays the information in a readable format. The RssAggregator Server Control lets you do the same by processing and displaying an RSS feed from any Web site. This allows you to create an information Web page for news, stories, images, lists of music, or for whatever purpose someone creates an RSS feed.

You can create a multi-level information Web page by using multiple RssAggregator Server Controls on the same page. For example, you could create a world news Web page by adding RSS feeds from the NY Times, The BBC, and AFP (Agence France-Presse).

The process of adding an RSS Aggregator to your Web site is simple.

1. Choose a Web form where the RSS Aggregator will reside
2. Drag and drop the RssAggregator Server Control on that Web form
3. Set the URL property to point at the RSS feed. For example, `http://msdn.microsoft.com/rss.xml`
4. Save the Web form

When a visitor to your site views the Web form in a browser, he sees the RSS feed displayed properly. When he refreshes the Web page, any information that the RSS feed provider has updated is now reflected. *See Also:* ”Using the RssAggregator Server Control” on page 196

By creating a custom XSLT, you can change the way an RSS feed is viewed by your site visitors.

This subsection contains the following topics:


- ”RSSAggregator Server Control Properties” on page 195
- ”Using the RssAggregator Server Control” on page 196

# RSSAggregator Server Control Properties

The RSSAggregator Server Control properties are described in this table.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| Authenticated | Indicates if you are logged in to the CMS Explorer and can use it to browse to Content, Collections, etc. *See Also:* "Using CMS Explorer to Browse Your Ektron CMS400.NET Site" on page 32 | String |
| DisplayXslt | Determines how the information on the page is displayed<br>**None**-databind only<br>**ecmNavigation** - lists the title of every RSS feed item<br>*See Also:* "Example of ecmNavigation Display" on page 122<br>**ecmTeaser** - lists a title and a description of every RSS feed item<br>*See Also:* "Example of ecmTeaser Display" on page 123<br>**Path to Custom Xslt** - Enter the path to an Xslt that determines the display of the page | String |
| Hide | Used to hide a RssAggregator Server Control in design time and run time.<br>**True** = Hide Metadata Server Control<br>**False** = Show Metadata Server Control | boolean |
| Language | Design time only: View content blocks, collections, etc. in different languages.<br>Note: The language property shows results only in design-time (in Visual Studio) and does not affect the site. Content on the site is managed by a cookie value that is set by selecting a language from the dropdown list. | Integer |

| Property | Value | Data Type |
|---|---|---|
| MaxResults | The Maximum number of items from an RSS feed that are returned (0=unlimited). | Integer |
| URL | The RSS feed path for the server control.<br>For example:<br>`http://msdn.microsoft.com/rss.xml` | String |
| WrapTag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an inline portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code.<br>**Custom** - Allows you to use a custom tag. | String |

# Using the RssAggregator Server Control

These steps show how to use the RssAggregator Server Control.

1. Drag a RssAggregator Server Control into a template.
2. Set the URL property with the path of the RSS feed you want to display. This example use the following RSS feed:

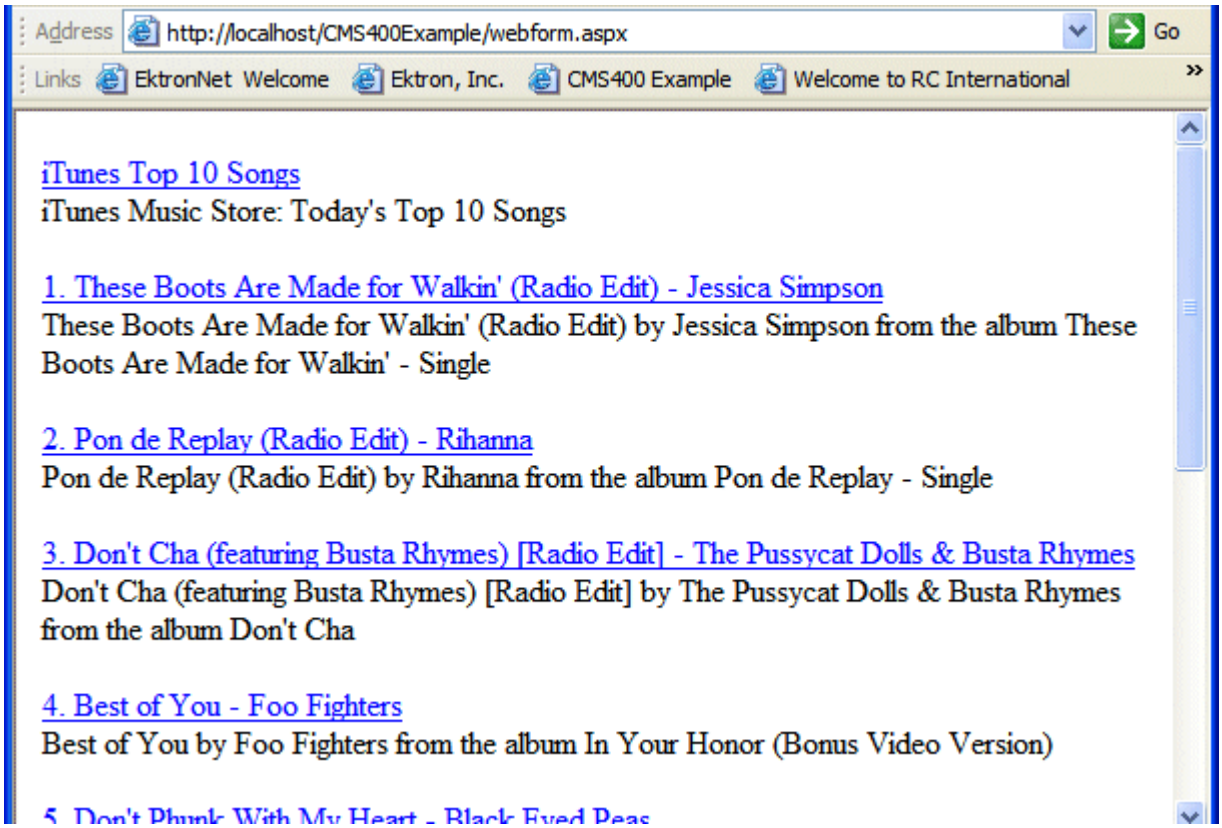
```
http://ax.phobos.apple.com.edgesuite.net/WebObjects/MZStore.woa/wpa/MRSS/topsongs/limit=10/rss.xml
```

   This feed displays the top 10 songs according to the iTunes Music Store.

**NOTE** In addition to the URL property, other Ektron properties are available to customize this server control. For more information on these properties, see "RSSAggregator Server Control Properties" on page 195

3. Build the Web form.

4. Browse to the Web page in your browser.


Address
http://localhost/CMS400Example/webform.aspx
Go
Links
EktronNet Welcome
Ektron, Inc.
CMS400 Example
Welcome to RC International
iTunes Top 10 Songs
iTunes Music Store: Today's Top 10 Songs
1. These Boots Are Made for Walkin' (Radio Edit) - Jessica Simpson
These Boots Are Made for Walkin' (Radio Edit) by Jessica Simpson from the album These Boots Are Made for Walkin' - Single
2. Pon de Replay (Radio Edit) - Rihanna
Pon de Replay (Radio Edit) by Rihanna from the album Pon de Replay - Single
3. Don't Cha (featuring Busta Rhymes) [Radio Edit] - The Pussycat Dolls & Busta Rhymes
Don't Cha (featuring Busta Rhymes) [Radio Edit] by The Pussycat Dolls & Busta Rhymes from the album Don't Cha
4. Best of You - Foo Fighters
Best of You by Foo Fighters from the album In Your Honor (Bonus Video Version)

5. The page appears with an RSS list of information.

# BreadCrumb Server Control

The BreadCrumb Server Control creates a breadcrumb trail of visited pages for your site visitors. This gives them the ability to navigate back to Web pages they have already visited.

The breadcrumb trail can appear vertically or horizontally on a Web page. You can define how many breadcrumbs are left for your Web site users to navigate back. You can also use an icon and Alt text, instead of using text only to display the breadcrumb trail.

| Breadcrumb Trail Type | Example |
|---|---|
| Horizontal with Text Only | Dev > WebForm > pr > FAQ > Meta* |
| Horizontal with Icons | |
| Horizontal with Icons and Alt Text | |
| Vertical with Text Only | > Dev<br>> WebForm<br>> pr<br>> FAQ<br>> Meta* |

| Breadcrumb Trail Type | Example |
|---|---|
| Vertical with Icons | |
| Vertical with Icons and Alt Text | |

A BreadCrumb Server Control should be added to every Web page in your Web site. This ensures that a breadcrumb trail is created for each Web page a user visits.

Each BreadCrumb Server Control on your site can be set up to individually. For example, you might want the breadcrumb trail appear horizontally on one Web page and vertically on another.

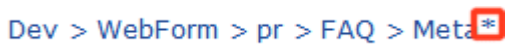
This subsection contains the following topics:


- ”BreadCrumb Server Control Properties” on page 199
- ”Using the BreadCrumb Server Control” on page 201


# BreadCrumb Server Control Properties

The BreadCrumb Server Control properties are described in this table.

**NOTE** The following table only lists Ektron-specific properties. It does not describe native .Net properties such as font, height, width and border style. For documentation of these properties, see VisualStudio.Net help.

| Property | Value | Data Type |
|---|---|---|
| CurrentPageIndicator | A symbol or character used to identify, in the breadcrumb trail, the page you are currently on.<br>  | String |
| DisplayStyle | Whether you want the breadcrumb trail to be displayed horizontally or vertically. *See Also:* "Breadcrumb Trail Type" on page 198 | String |
| DisplayTitle | What the breadcrumb shows for a page name.<br>The default = **Title**. | String |
| IconAlt | The Alt text that appears when the icon is hovered over. | String |
| IconPath | The path to the icon image. This icon replaces the text that appears in the DisplayTitle property.<br>For example: `Images\bc_meta_icon.gif`<br>You do not need to define the site root in the property.<br>**Important:** The image location must be relative to the site root. | String |
| LinkLastItem | Choose whether the last breadcrumb item appears as a hyperlink.<br>**True** = Last item as hyperlink<br>**False** = Last item as test | Boolean |
| LinkTarget | Set whether the link opens up in the same window, a parent window, a new window, or a parent frame, when a user clicks a breadcrumb link.<br>The default = **_self**<br>**_Self** = Opens in the same window<br>**_Top** = Opens in parent window<br>**_Blank** = Opens in new window<br>**_Parent** = opens in the parent frame. | String |

| Property | Value | Data Type |
|---|---|---|
| MaxItems | The maximum amount of items displayed in the breadcrumb trail.<br>The default = **5**<br>**Zero (0)** = Unlimited breadcrumb items. | Integer |
| Separator | What separates the items in the breadcrumb trail.<br>The default = **> (greater than)**<br> | String |

# Using the BreadCrumb Server Control

To use the BreadCrumb Server Control, you need to add it to each Web form for which you want to create a breadcrumb trail.

These steps show how to use the BreadCrumb Server Control.

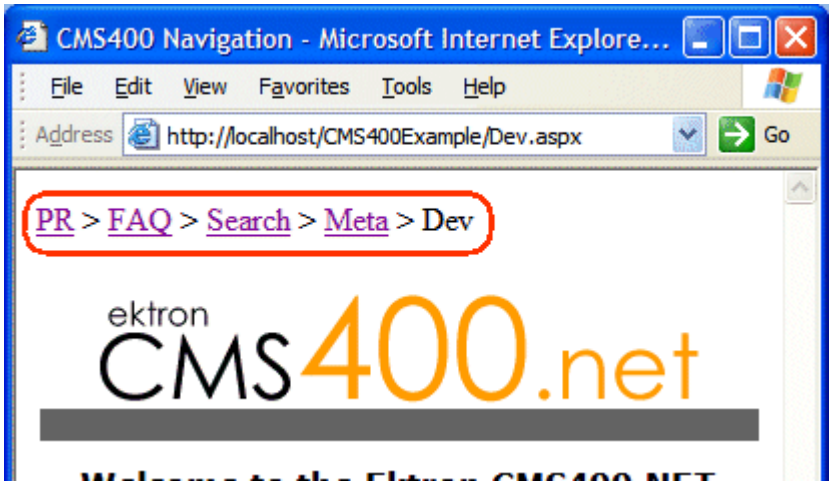
1. Drag an drop the BreadCrumb Server Control onto a Web form for which you want to create a breadcrumb trail.
2. Set the following property

   **DisplayTitle** - Choose a name for the Web form that appears in the breadcrumb trail, when a user views the web page.

**NOTE** If you would like to customize the look of your breadcrumb trail, you can do so by configuring any of the other properties. For more information on the BreadCrumb Server Control properties. see"BreadCrumb Server Control Properties" on page 199.

3. Save the Web form.
4. Repeat steps one through three for all the Web forms to be included in the breadcrumb trail.
5. Open a browser.
6. View a Web page with a BreadCrumb Server Control in it.
7. Navigate to the rest of the Web pages that contain BreadCrumb Server Controls.

8. As you move around, the names of the pages are added to the breadcrumb trail.



# Dreamweaver Extension Support

Ektron CMS400.NET has a custom Macromedia® Extension Package, which allows you to not only create your templates in Dreamweaver® or UltraDev™, but also to insert the custom functions through Dreamweaver®.

To do this, all you need is Ektron CMS400.NET, Dreamweaver®/ UltraDev™, and to keep reading.

The following sections are contained in this chapter:


- ”Requirements” on page 203
- ”Preparing Dreamweaver for Ektron CMS400.NET Functions” on page 203
- ”Installing the Extension” on page 204
- ”Setup” on page 208
- ”Using the Ektron CMS400.NET Extension” on page 210
- ”Inserting Custom Functions” on page 216


## Requirements

To use the Dreamweaver extension supplied by Ektron, you need to have Dreamweaver 7.0.1 or higher installed, as well as Ektron CMS400.NET.

## Preparing Dreamweaver for Ektron CMS400.NET Functions

To have **Ektron CMS400.NET** function symbols appear in Dreamweaver, while editing a Web page, you must edit two `PreviewModeCustomTags.edml` Dreamweaver files. The steps below explain how to edit the files.

1. In Windows Explorer navigate to the following location

```
C:\Program Files\Macromedia\Dreamweaver MX 2004\Configuration\Translators\ASP.NET_VB\
```

2. Open the `PreviewModeCustomTags.edml` file using Notepad.
3. Search for the following line of code in the file.

```
<searchPattern requiredLocation="openTag"><![CDATA[/runat\s*=\s*"?server"?/i]]></
searchPattern>
```

4. Add the following line after it.

```
<searchPattern requiredLocation="tagName"><![CDATA[/^(?!cms:)/i]]></searchPattern>
```

**WARNING!** This line must be inserted after the line above. Order is important.

5. Save and close the file.
6. Navigate to the following location.

```
C:\Program Files\Macromedia\Dreamweaver MX 2004\Configuration\Translators\ASP.NET_Csharp\
```

7. Open the `PreviewModeCustomTags.edml` file using Notepad.
8. Search for the following line of code in the file.

```
<searchPattern requiredLocation="openTag"><![CDATA[/runat\s*=\s*"?server"?/i]]></
searchPattern>
```

9. Add the following line after it.

```
<searchPattern requiredLocation="tagName"><![CDATA[/^(?!cms:)/i]]></searchPattern>
```

**WARNING!** This line must be inserted after the line above. Order is important.

10. Save the file.

You are now ready to Install the **Ektron CMS400.NET** extension.
*See Also:* ”Installing the Extension” on page 204

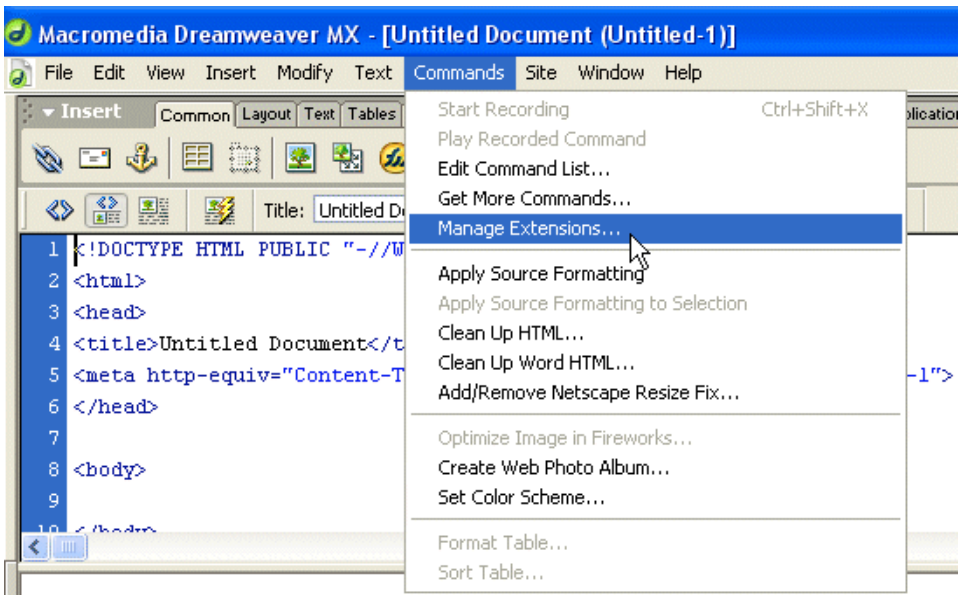
# Installing the Extension

Before you can insert the Ektron CMS400.NET custom functions through Dreamweaver®, you must install the extension. Listed below are the steps to install the extension.

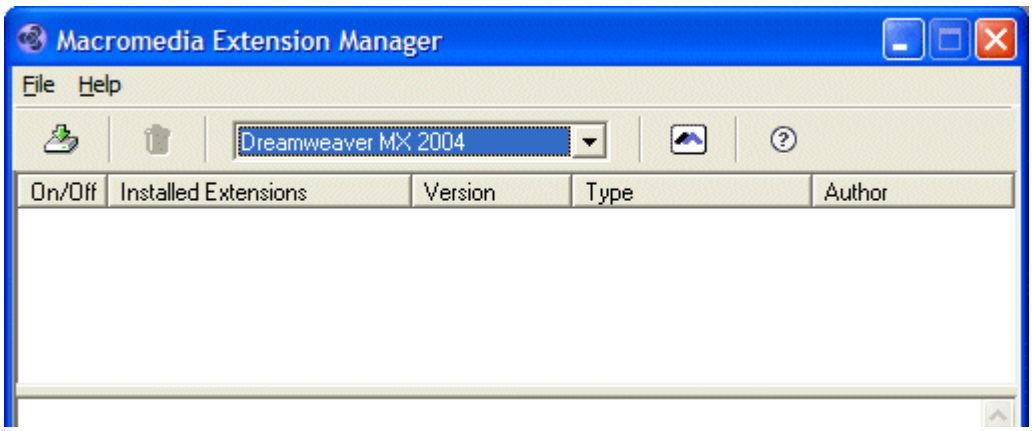
**NOTE** For additional information about installing an extension, refer to your Dreamweaver® Manual.

To Install the extension, follow these steps.

1. Begin by following the path in Dreamweaver:
   **Commands > Manage Extensions...**

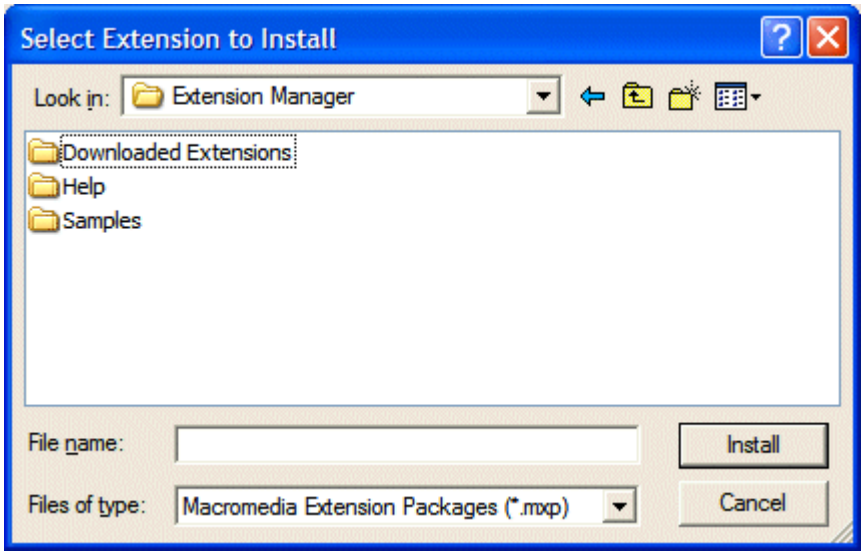



The Macromedia Extension Manager is displayed.



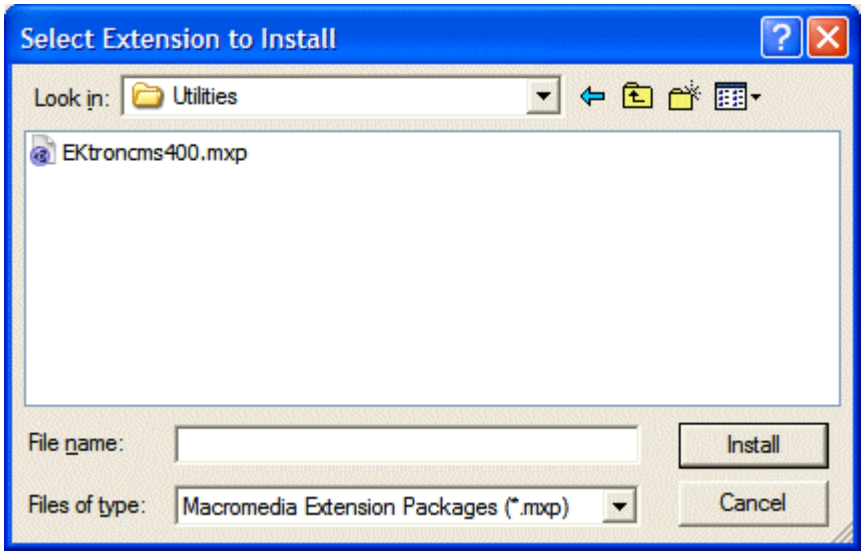

2. From this dialog box, follow the path:
   **File > Install Extension...**

3. The Select Extension to Install window opens.



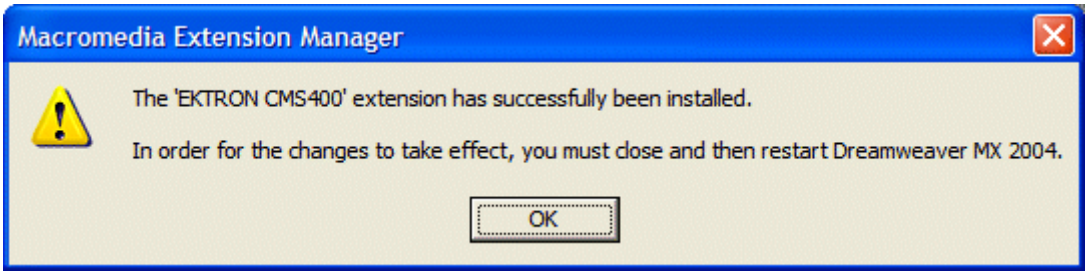

4. Locate the Macromedia® Extension Package (.mxp) file that corresponds to Ektron CMS400.NET. By default, the file gets installed to the following directory:

   **C:\Program Files\Ektron\CMS400v50\Utilities**

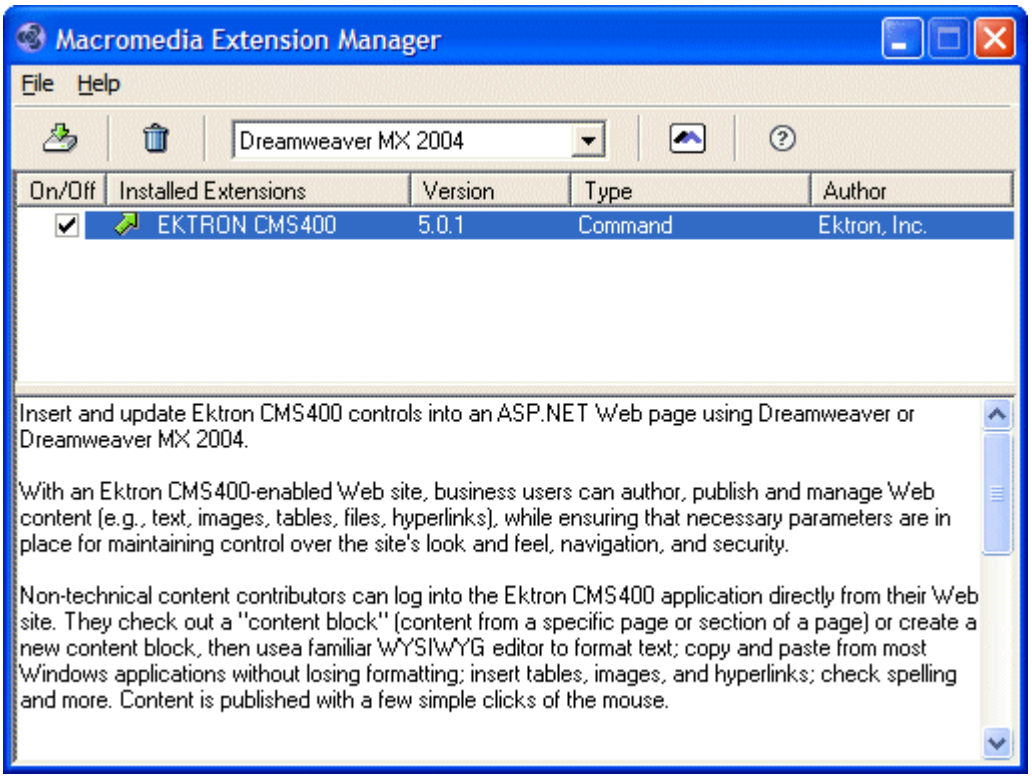



5. After you select the file, click the **Install** button.
6. The installation begins with the Macromedia® Extensions Disclaimer. Click **Accept**.

7. The extension begins to install. Once installed, you see a confirmation box.


Macromedia Extension Manager
The 'EKTRON CMS400' extension has successfully been installed.
In order for the changes to take effect, you must close and then restart Dreamweaver MX 2004.
OK

8. Click **OK** to continue.
9. The Macromedia Extension Manager is again displayed, but with the added extension.


Macromedia Extension Manager
File Help
Dreamweaver MX 2004
On/Off
Installed Extensions
Version
Type
Author
EKTRON CMS400
5.0.1
Command
Ektron, Inc.
Insert and update Ektron CMS400 controls into an ASP.NET Web page using Dreamweaver or Dreamweaver MX 2004.
With an Ektron CMS400-enabled Web site, business users can author, publish and manage Web content (e.g., text, images, tables, files, hyperlinks), while ensuring that necessary parameters are in place for maintaining control over the site's look and feel, navigation, and security.
Non-technical content contributors can log into the Ektron CMS400 application directly from their Web site. They check out a "content block" (content from a specific page or section of a page) or create a new content block, then usea familiar WYSIWYG editor to format text; copy and paste from most Windows applications without losing formatting; insert tables, images, and hyperlinks; check spelling and more. Content is published with a few simple clicks of the mouse.

10. If needed, exit Dreamweaver®, then start it back up.

Now you can use the Ektron CMS400.NET Extension Pack.

Refer to "Using the Ektron CMS400.NET Extension" on page 210 for more information on using Dreamweaver® to create and edit templates for Ektron CMS400.NET.

# Setup

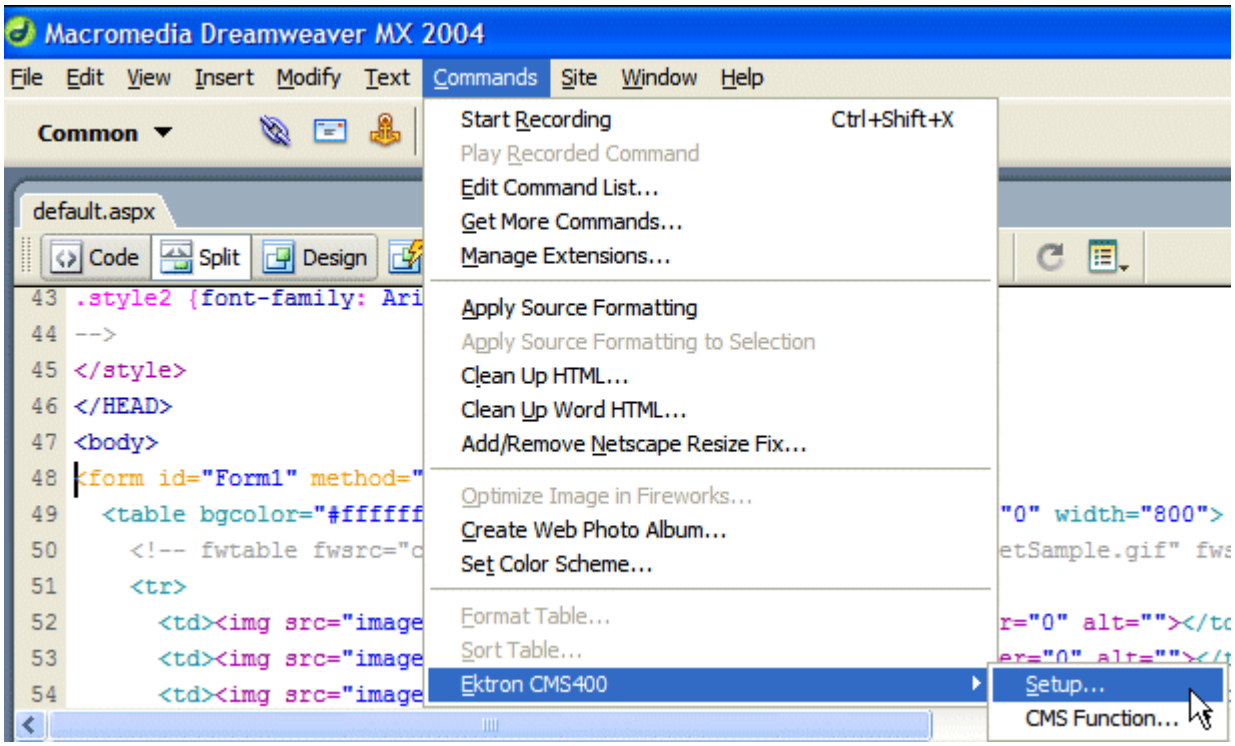
Before you can use Dreamweaver®/UltraDev™ to create and edit Ektron CMS400.NET templates, you should configure the Ektron CMS400.NET Extension. Listed below are the configurative options for the extension in Dreamweaver:

- Username
- Password
- Domain
- Server File
- Server
- Work Online
- Include
- Check for Include

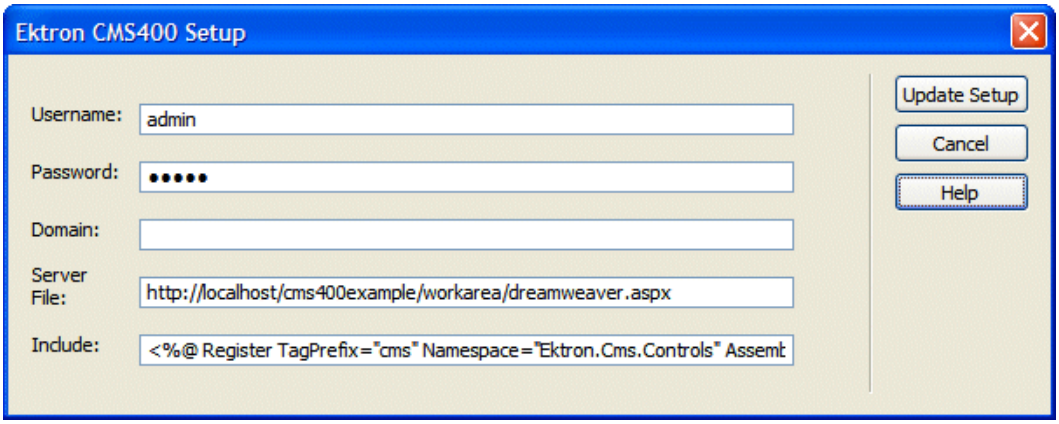
To change these options, follow these steps.

1. Follow the path:

   **Commands > Ektron CMS400.NET > Setup...**



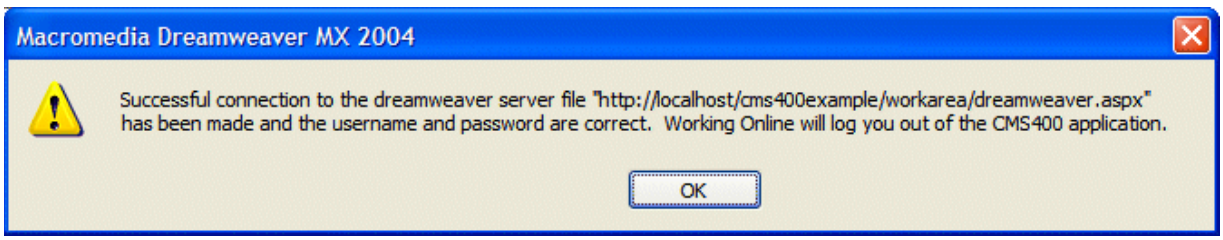

The Ektron CMS400.NET Setup dialog box is displayed on the screen.



2. Use the following table to help you update your setup information.

| Field | Description |
|---|---|
| Username | Valid username of a user of your Ektron CMS400.NET Web site. When using the extension, the username is used to log in to the database to retrieve information about content blocks and other Web site information |
| Password | Enter the password for the user. |
| Domain | Only required if Active Directory has been enabled for your Web site. Enter the domain corresponding to the username entered. |
| Server File | Specify the location and file name for the dreamweaver.aspx file needed for the extension to properly work and make connection with your Ektron CMS400.NET database. |
| Include | Each Ektron CMS400.NET template requires an include file to point to the API with the definitions for the custom functions. Here, specify the include file. |

3. Click Update Setup.
4. A message appears.


Macromedia Dreamweaver MX 2004
Successful connection to the dreamweaver server file "http://localhost/cms400example/workarea/dreamweaver.aspx" has been made and the username and password are correct. Working Online will log you out of the CMS400 application.
OK

**NOTE** A message other than this is probably due to an incorrect server file, username, or password. Recheck your settings and try again.

5. Click **OK** to continue.
6. The setup dialog box closes. You are ready to create templates using Macromedia Dreamweaver.

# Using the Ektron CMS400.NET Extension

Now that the Macromedia® Extension Package has been installed for Dreamweaver, you can create and edit templates for Ektron CMS400.NET.

To explain how to use this, we use templates that are in the Ektron CMS400.NET installation, but with the custom functions stripped out.

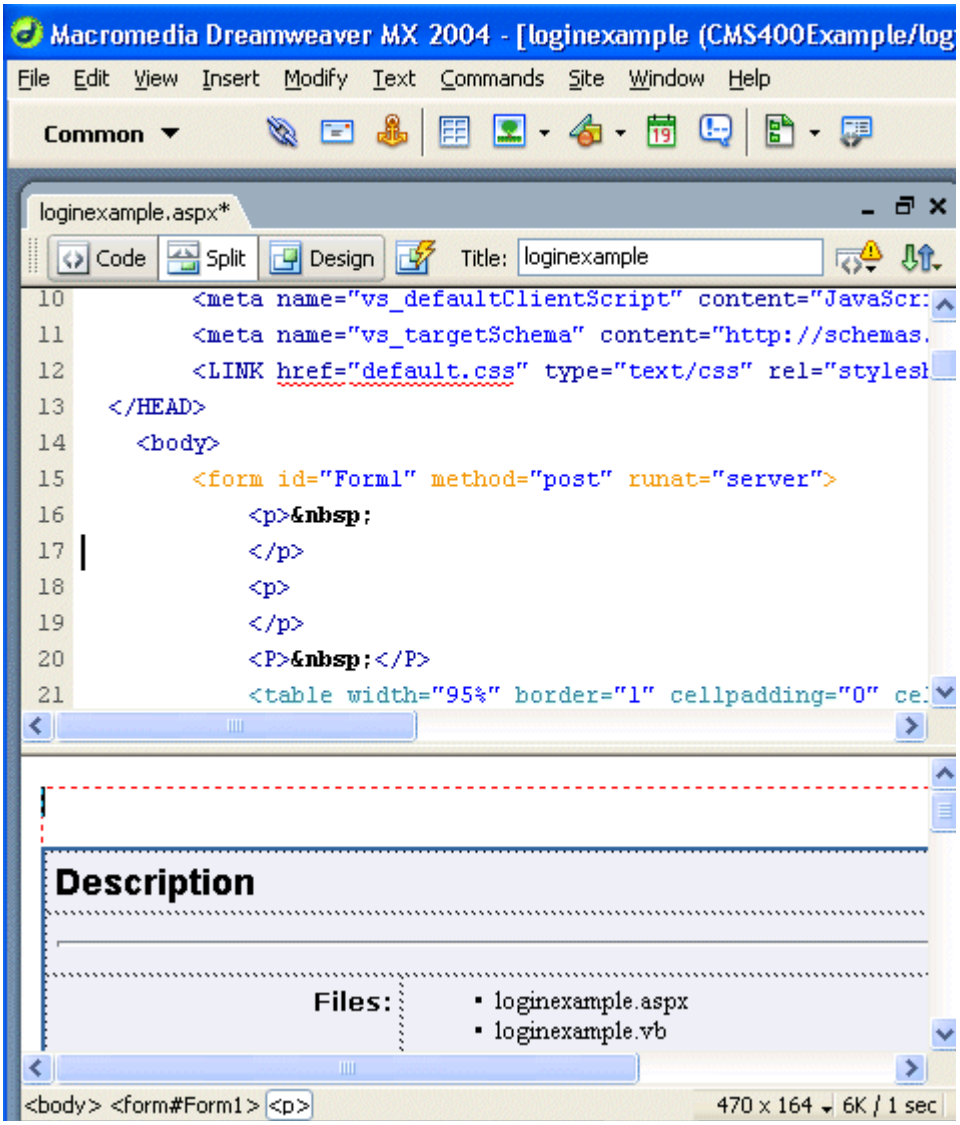
**NOTE** You can also create your own templates instead of using the ones in the Ektron CMS400.NET installation.

The following subsections are contained in this section:


- ”Custom Function Symbols” on page 213
- ”Ektron CMS400.NET Commands Menu” on page 214

In Dreamweaver, begin by opening the file:

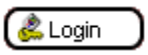
**\webroot\CMS400Example\loginexample.aspx**

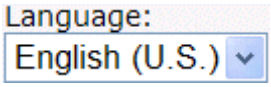



**NOTE** The custom ASPX function that is inserted into this template by default has been stripped out.

## Custom Function Symbols

During the process of inserting functions, symbols are used to represent the custom display functions. The following table explains each symbol.

| Symbol | Represents | More Information |
|---|---|---|
|  | Login/Logout button | ”Login” on page 217 |
| | Static content block. | ”Inserting a Content Block” on page 220 |
| | Search | ”Inserting a Search Function” on page 232 |
| | List Summary | ”Inserting a List Summary” on page 225 |
| No symbol appears when Metadata is inserted in a Web page. | Metadata | ”Inserting a Metadata Function” on page 229 |
| | Collection | ”Inserting a Collection” on page 222 |
| | Event Calendar | ”Inserting an Event Calendar Function” on page 235 |
| | Form Content Block | ”Inserting a Form” on page 238 |

| Symbol | Represents | More Information |
|---|---|---|
| | XML Index Search | "Inserting an XML Index Search" on page 241 |
| | Display By Meta Value | "Inserting Display By Meta Value" on page 246 |
| | Standard Menu | "Inserting a Standard Menu" on page 250 |
| | DHTML Menu | "Inserting a DHTML Menu" on page 253 |
|  | Language Select | "Inserting a Language Select Box" on page 258 |
| | Language API | "Inserting a Language API" on page 260 |

## Ektron CMS400.NET Commands Menu

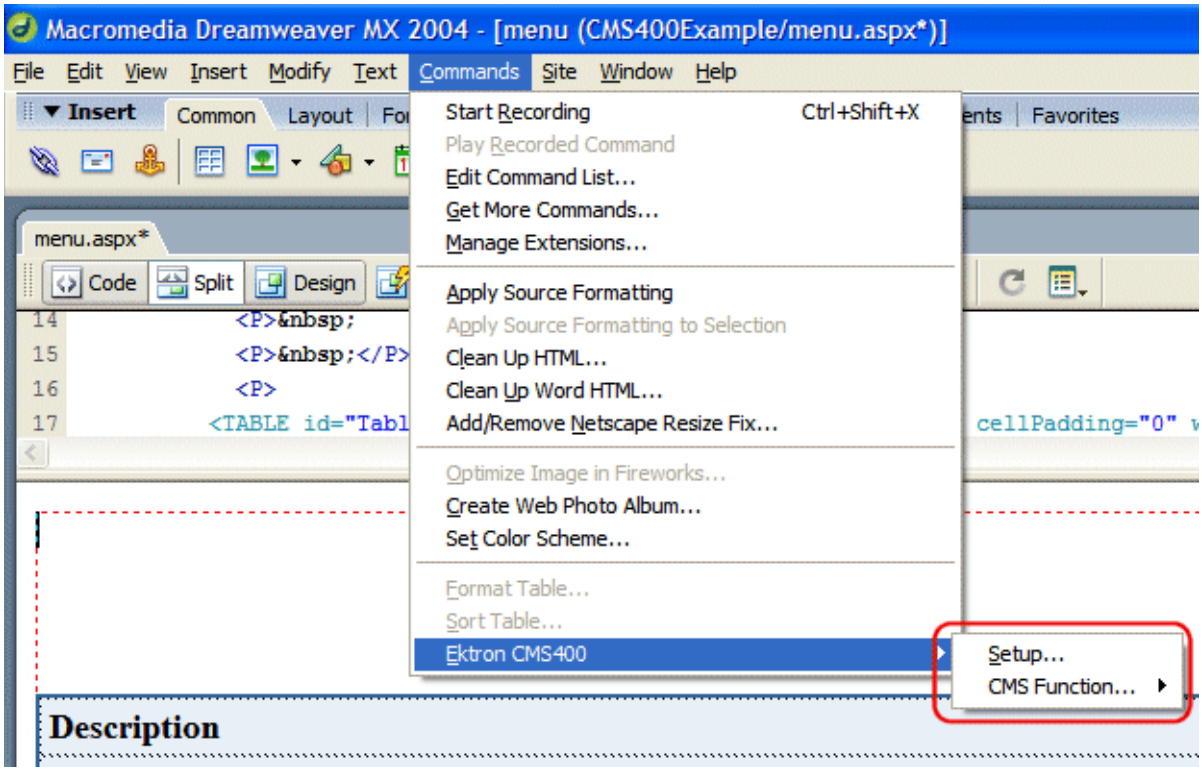
The following section explains how to locate Ektron CMS400.NET commands in Dreamweaver.

To locate the command menu in Dreamweaver, perform the following steps:

1. Open Dreamweaver.
2. In the top menu, follow the path:

   **Commands > Ektron CMS400.NET**

3. The Ektron CMS400.NET commands menu is displayed.




4. Click on the command for the task you would like to perform.

   The following table explains each command.

| Command | Description | More Information |
|---|---|---|
| Setup | Opens Ektron CMS400.NET Setup information dialog for Dreamweaver. | "Setup" on page 208 |
| **CMS Function...** | | |
| Login | Inserts an Ektron CMS400.NET login function into the template. | "Login" on page 217 |
| Content | Inserts an Ektron CMS400.NET content block function into the template. | "Inserting a Content Block" on page 220 |
| Collection | Inserts an Ektron CMS400.NET collection function into the template. | "Inserting a Collection" on page 222 |
| List Summary | Inserts an Ektron CMS400.NET list summary function into the template. | "Inserting a List Summary" on page 225 |

| Command | Description | More Information |
|---|---|---|
| Metadata | Inserts an Ektron CMS400.NET metadata function into the template. | ”Inserting a Metadata Function” on page 229 |
| Search | Inserts an Ektron CMS400.NET search function into the template. | ”Inserting a Search Function” on page 232 |
| Calendar | Inserts an Ektron CMS400.NET event calendar function into the template. | ”Inserting an Event Calendar Function” on page 235 |
| Form | Inserts an Ektron CMS400.NET form content block into the template. | ”Inserting a Form” on page 238 |
| XML Index Search | Inserts an Ektron CMS400.NET XML Index Search into the template. | ”Inserting an XML Index Search” on page 241 |
| Display By Meta Value | Inserts an Ektron CMS400.NET display by meta value block into the template. | ”Inserting Display By Meta Value” on page 246 |
| Standard Menu | Inserts an Ektron CMS400.NET standard menu into the template. | ”Inserting a Standard Menu” on page 250 |
| DHTML Menu | Inserts an Ektron CMS400.NET DHTML menu into the template. | ”Inserting a DHTML Menu” on page 253 |
| Language Select | Inserts an Ektron CMS400.NET language select box into the template. | ”Inserting a Language Select Box” on page 258 |
| Language API | Inserts an Ektron CMS400.NET Language API into the template. | ”Inserting a Language API” on page 260 |

# Inserting Custom Functions

## CMS Function

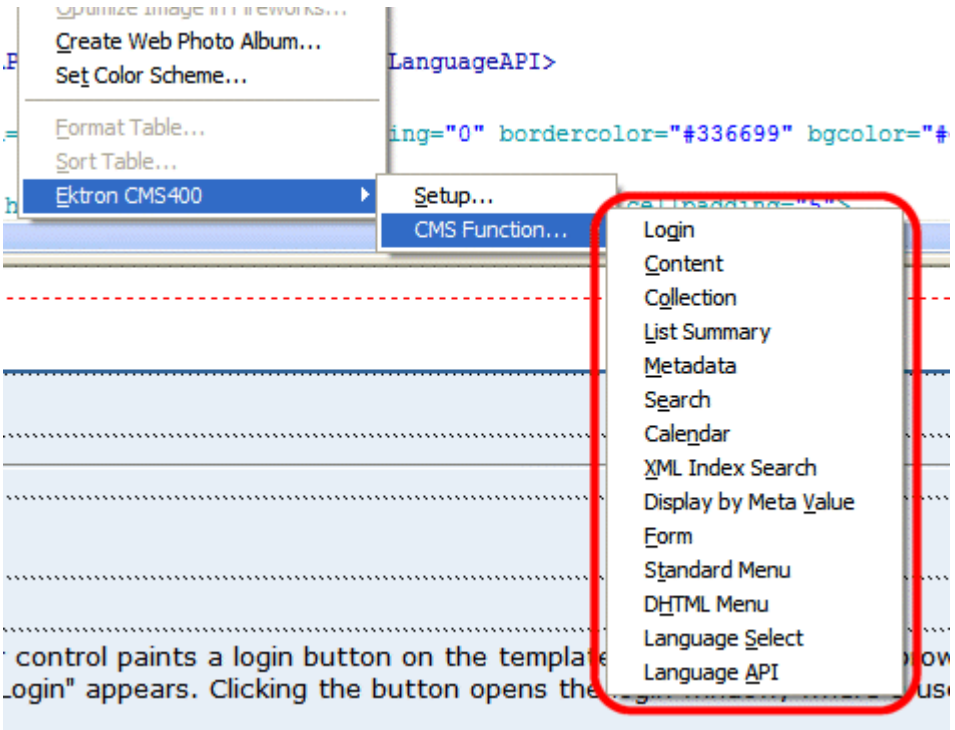
The CMS Function commands let you insert standard Ektron CMS400.NET display functions into your template.

## Accessing CMS Content Functions

To access the CMS Content commands menu, follow this path:

**Commands > Ektron CMS400.NET... > CMS Function...**

A list of all CMS Content functions that can be inserted is displayed.




The following sections explain how to insert each function.

## Login

For a description of the Login button, see "Login Server Control" on page 63.

To insert a login button:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET functions, click **Login**.

3. The Insert Ektron CMS400.NET Login box appears.

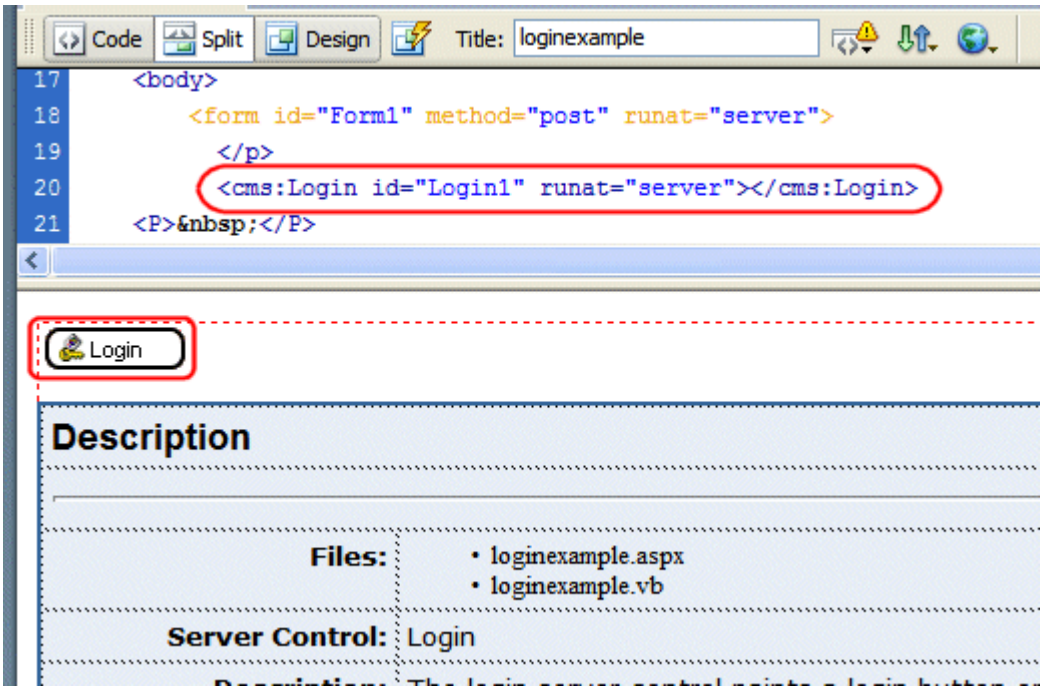
4. Use the following table to assist you with inserting a Login button.

| Field | Description |
|---|---|
| ID | A unique name for the function you are inserting. |
| Hide | Used to hide a Login button at run time.<br>**Checked** = Hide Login button<br>**Unchecked** = Show Login button<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page.<br>For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |

| Field | Description |
|---|---|
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |

5. Click **insert**.

   A login icon indicates where the Login button appears on the template.




Shown below is an example of the code that is inserted:

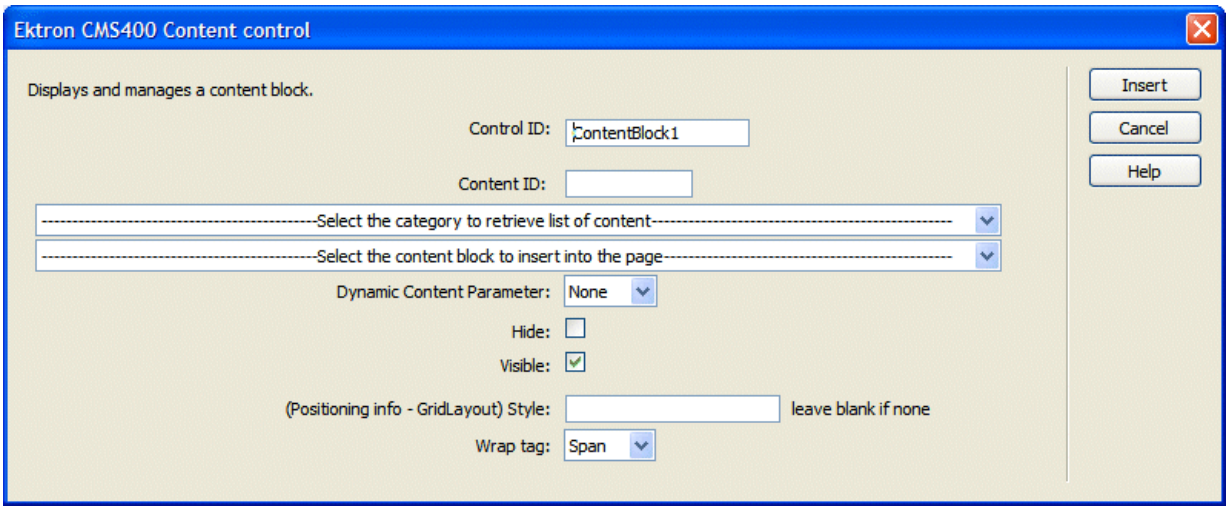
| Platform | Code View |
|---|---|
| .NET | <cms:Login id="Login1" runat="server"></cms:Login> |

## Inserting a Content Block

For a description of the Content Block function, see "ContentBlock Server Control" on page 71.

To insert a content block:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Content**.
   The Insert Content box is displayed




3. Use the following table to help you insert a content block.

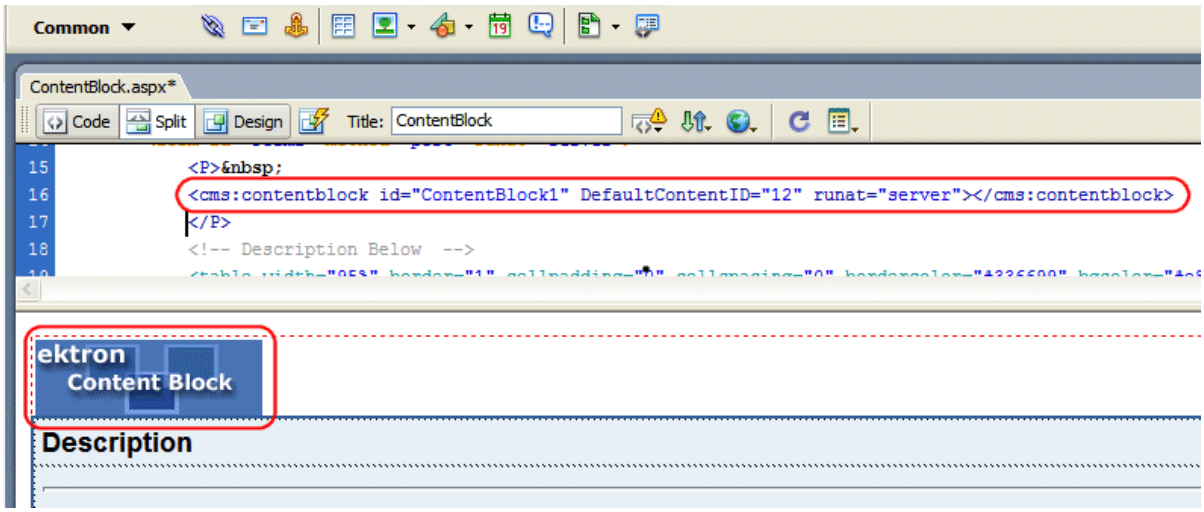
| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| Content ID | Specify the ID number of the content block that you wish to use as a content block on the Web page. If you do not know the Content ID, you can navigate to it using the Select Category and Select Content drop down boxes. |
| Select Category | Choose the folder where the content is located. If you insert the content ID, you do not need to select a category. |
| Select Content | Select the content block to use from the list. If you insert the content ID, you do not need to select content. |

| Field | Description |
|---|---|
| Dynamic Content Parameter | Determines if content is displayed as static or passed dynamically. By making content dynamic, when the content displayed on a Web page contains a link to another piece of content, you can display the second piece of content in the same Web page space.<br>**None**: Displays a static content block.<br>**id**: Content block is passed as a URL parameter. |
| Hide | Used to hide a content block in design time and run time.<br>**True** = Hide content block<br>**False** = Show content block<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page.<br>For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |

**NOTE** You see only the list of Ektron CMS400.NET content blocks if you are working online.

4. Click **insert**.

   A graphic indicates the static content block's location on the template.




Shown below is an example of the code that is inserted.

| Platform | Code View |
|---|---|
| .NET | <cms:contentblock id="ContentBlock1" DefaultContentID="12" runat="server"></cms:contentblock> |

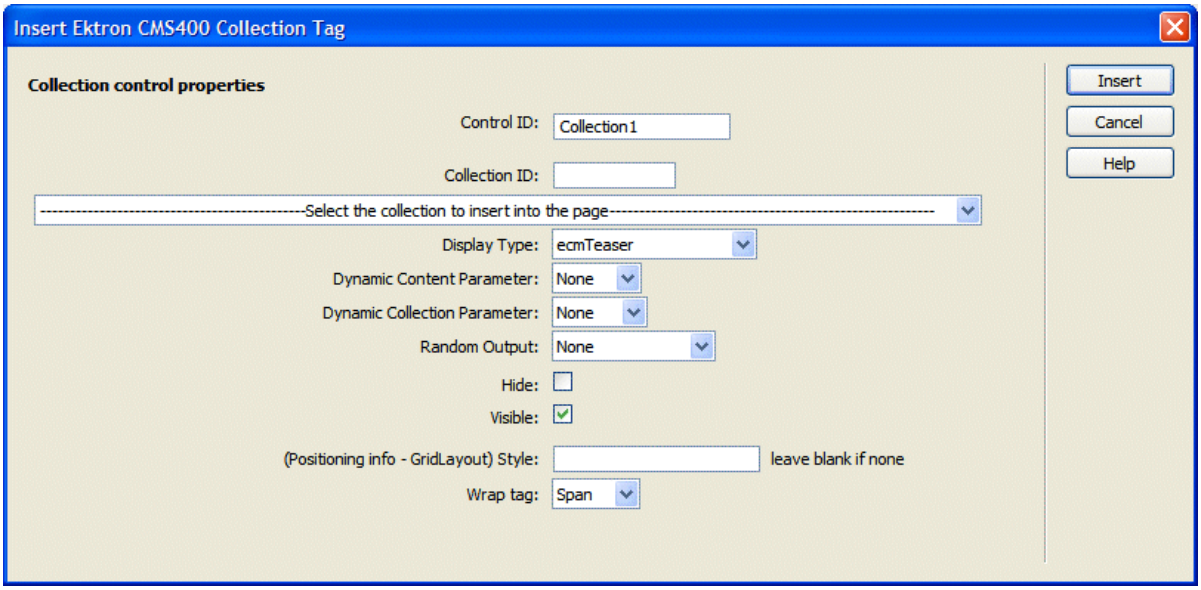
## Inserting a Collection

For a description of the Collection function, see "Collection Server Control" on page 120

To insert a collection:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Collection**.

   The Insert Ektron Collection box is displayed.



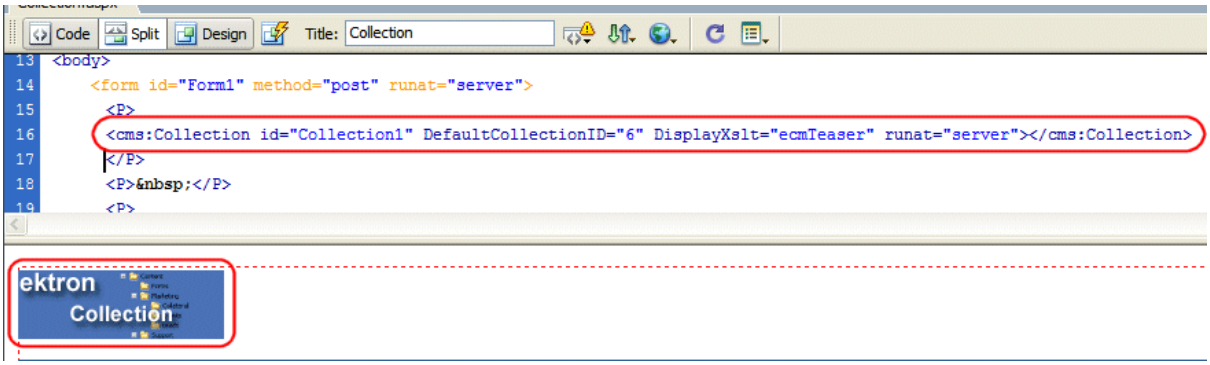

3. Use the following table to assist you with inserting a Collection.

| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| Collection ID | If you know the collection ID, enter it into the textbox. If you do not know the Collection ID, you can navigate to it using the Select Collection drop down box. |
| Select Collection | The drop-down list is filled with a list of all the Collection titles that you have privileges to. When a title is selected, the Collection ID textbox is filled with that Collection's id number. If you insert the Collection ID, you do not need to select a category. |
| Display Type | Determines how the information on the page is displayed.<br>**None** - databind only<br>**ecmNavigation** - lists the title of every content block in the collection<br>*See Also:* "Example of ecmNavigation Display" on page 122<br>**ecmTeaser** - lists the title of every content block in the collection plus the content summary<br>*See Also:* "Example of ecmTeaser Display" on page 123 |

| Field | Description |
|---|---|
| Dynamic Content Parameter | Checks the QueryString for this value and replaces the collection with a content block when specified. Leave blank to always display the collection. |
| Dynamic Collection Parameter | To make this collection dynamic, select **coll_id**. When you do, the function uses the collection passed as a URL parameter. |
| Random Output | Displays random content within a collection.<br>**None** - Displays all content in collection.<br>**Random Teaser** - Displays a random content block's teaser.<br>**Random Body** - Displays a random content block's body. |
| Hide | Used to hide a collection in design time and run time.<br>**Checked** = Hide collection<br>**Unchecked** = Show collection<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page.<br>For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |

4. When completed, click **insert**.

   A graphic defines where the collection appears on your template.




Shown below is an example of the code that is inserted.

| Platform | Code View |
|---|---|
| .NET | `<cms:Collection id=”Collection1” DefaultCollectionID=”0” DisplayXslt=”ecmTeaser” runat=”server”></cms:Collection>` |

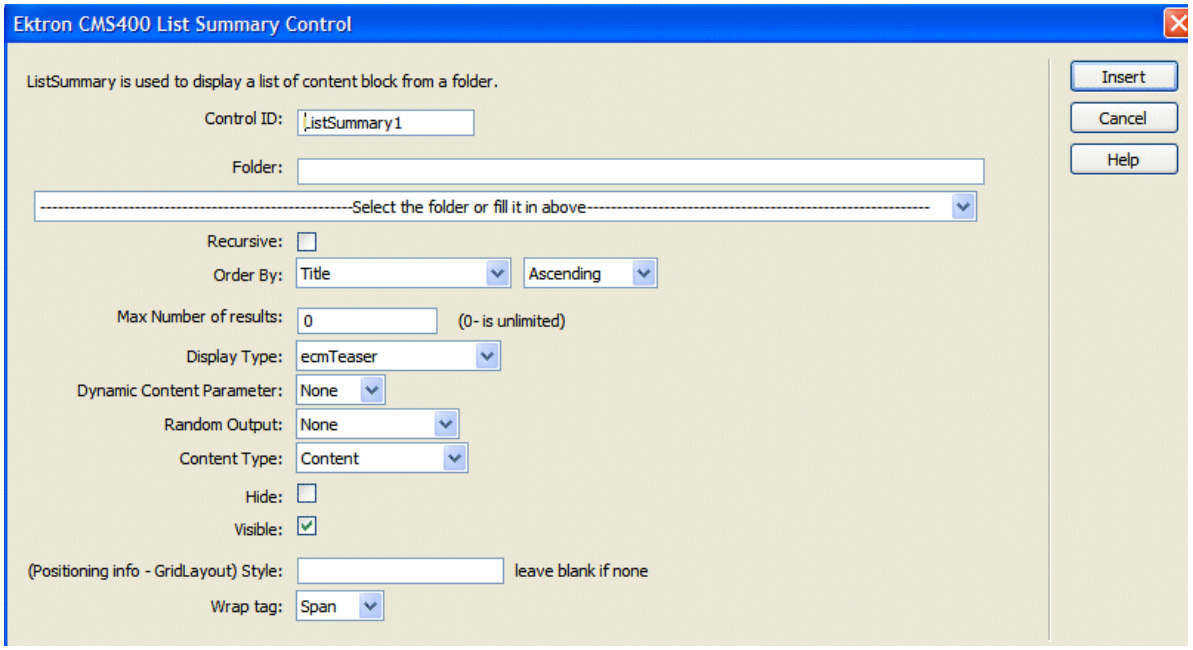
## Inserting a List Summary

For a description of the List Summary function, see ”ListSummary Server Control” on page 114.

To insert a list summary:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **List Summary**.

   The Insert Ektron List Summary box is displayed.




3. Use the following table to assist you with inserting a metadata control.

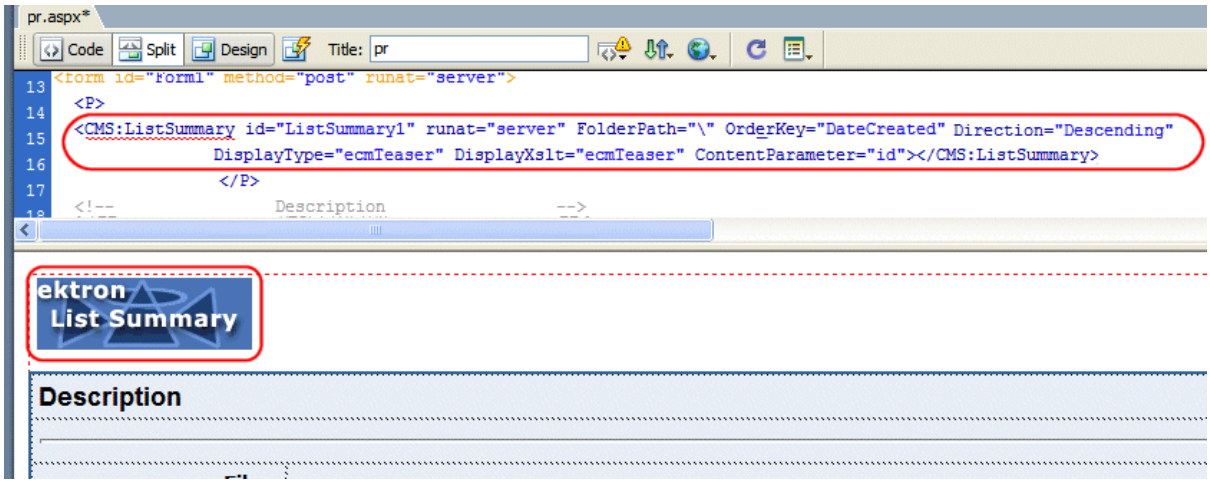
| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| Folder | Enter the folder name manually. The default is the root or "\". If you do not know the folder path, you can navigate to it using the Select folder drop down box. |
| Select Folder | Select the folder the summary information comes from. The folder drop-down contains folders from your site. If you insert the folder path, you do not need to select a folder from the dropdown box. |
| Recursive | If checked, summary information is gathered from the folder selected and all of its child folders. |

| Field | Description |
|---|---|
| Order By | Specify how the summaries are ordered. Sort by the following criteria.<br>• **Title** - The title of the content block<br>• **Date Created** - The date the content block was created<br>• **Date Modified** - The date the content block was last modified<br>• **Last Author Last Name** - The last author's last name<br>• **Last Author First Name** - The last author's first name<br>• **Ascending** - Alphabetic or numerical order<br>• **Descending** - Reverse alphabetic or numerical order |
| Max Number of results | Enter a number to specify how may results are displayed on the page (0=unlimited). |
| Display Type | Determines how the information on the page is displayed.<br>**None**-databind only<br>**ecmNavigation** - lists the title of every content block in the folder<br>*See Also:* "Example of ecmNavigation Display" on page 122<br>**ecmTeaser** - lists the title of every content block in the folder plus the content summary<br>*See Also:* "Example of ecmTeaser Display" on page 123 |
| Dynamic Content Parameter | Checks the QueryString for this value and replaces the list summary with a content block when specified. Choose **None** to always display List Summary. |
| Random Output | Displays random content within a list summary.<br>**None** - Displays all content in list summary.<br>**Random Teaser** - Displays a random content block's teaser.<br>**Random Body** - Displays a random content block's body. |
| Content Type | Defines what type of list summary appears.<br>**Content** - Displays only content blocks in the folder.<br>**All Types** - Displays all content in the folder.<br>**Forms** - Displays only forms in the folder.<br>**Archive_Content** - Displays archived content from the folder.<br>**Archive_Forms** - Displays archived forms from the folder. |

| Field | Description |
|---|---|
| Hide | Used to hide a list summary in design time and run time.<br>**Checked** = Hide list summary<br>**Unchecked** = Show list summary<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page.<br>For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |

4. When completed, click **insert**.

   A graphic defines where the list summary appears on your template.




Shown below is an example of the code that is inserted:

| Platform | Code View |
|---|---|
| .NET | `<CMS:ListSummary id="ListSummary1" runat="server" FolderPath="\" OrderKey="DateCreated" Direction="Descending" DisplayType="ecmTeaser" DisplayXslt="ecmTeaser" ContentParameter="id"></CMS:ListSummary>` |

## Inserting a Metadata Function

For a description of the Metadata function, see "MetaData Server Control" on page 186.

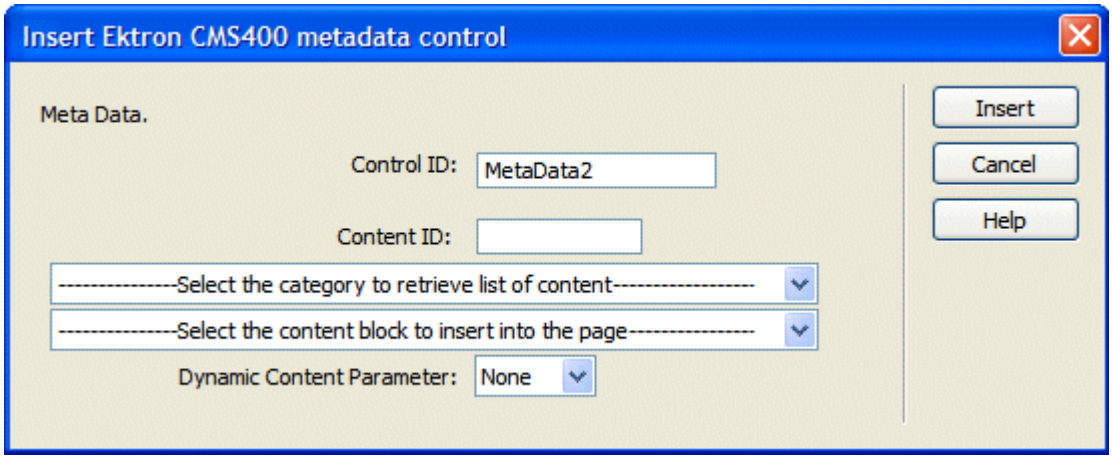
To insert a metadata function:

1. Move the blinking cursor to the area that you want to place it.

**NOTE** The Ektron CMS400.NET Metadata function should be inserted within the <head> tags of your template to work properly.

2. From the list of Ektron CMS400.NET Functions, click on **Metadata**.

   The Insert Ektron Metadata box is displayed.




3. Use the following table to assist you with inserting metadata.

| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| Content ID | Enter a valid content ID in the Content ID textbox. If you do not know the Content ID, you can navigate to it using the Select Category and Select Content Block drop down boxes. |
| Select Category | The drop-down list contains a list of content folders. If you insert the content ID, you do not need to select a category. |
| Select Content Block | The drop-down box is populated with a list of all the content block titles within that folder. When a title is selected, the content ID textbox is filled with that content block's id number. If you insert the content ID, you do not need to select a content block. |
| Dynamic Content Parameter | Gets or sets the QueryString parameter to read a content ID dynamically. |

4. When completed, click **insert**.
5. The following line appears in the <body> of the HTML code.

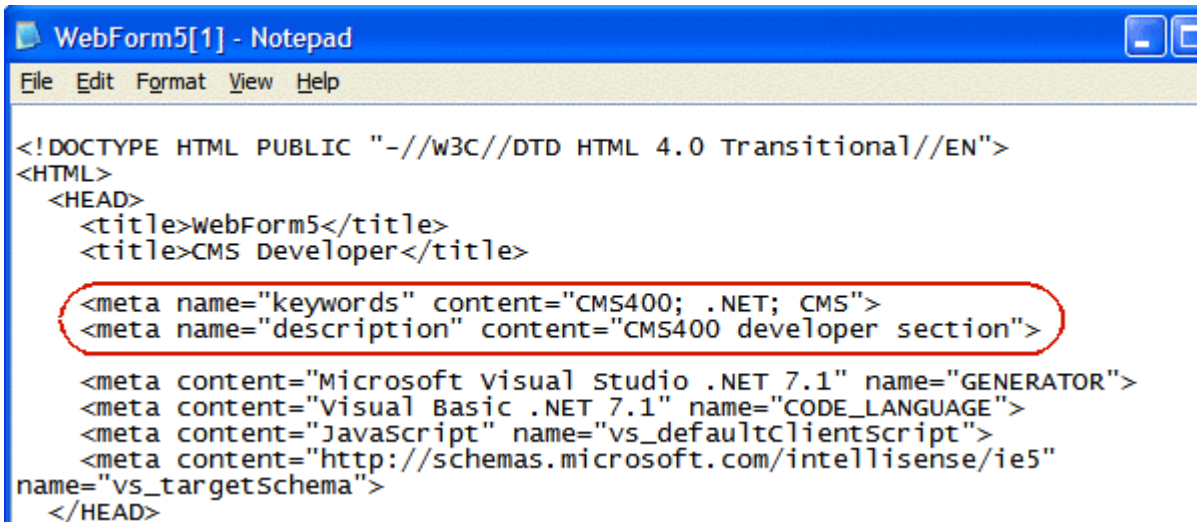
```
    <cms:MetaData id="MetaData2" runat="server"
DefaultContentID="12"></cms:MetaData>
```

6. Copy that line from the <body> tag into the <head> tag.

```
<HEAD>
    <title>WebForm5</title>
    <cms:MetaData id="Metadata2" runat="server" DefaultContentID="12"></cms:MetaData>
    <meta name="GENERATOR" content="Microsoft Visual Studio .NET 7.1">
    <meta name="CODE_LANGUAGE" content="Visual Basic .NET 7.1">
    <meta name="vs_defaultClientScript" content="JavaScript">
    <meta name="vs_targetSchema" content="http://schemas.microsoft.com/intellisense/i
</HEAD>
<body>
    <form id="Form1" method="post" runat="server">
    </form>
</body>
```

7. Save the Web form.
8. Open the Web page in the browser.
9. Right click on the Web page and click **View Source**. Look in the head tag. The meta tags from the content block are added to the page.

The metadata information added is shown below.




Shown below is an example of the code that is inserted:

| Platform | Code View |
|---|---|
| .NET | `<cms:MetaData id="MetaData2" runat="server" DefaultContentID="12"></ cms:MetaData>` |

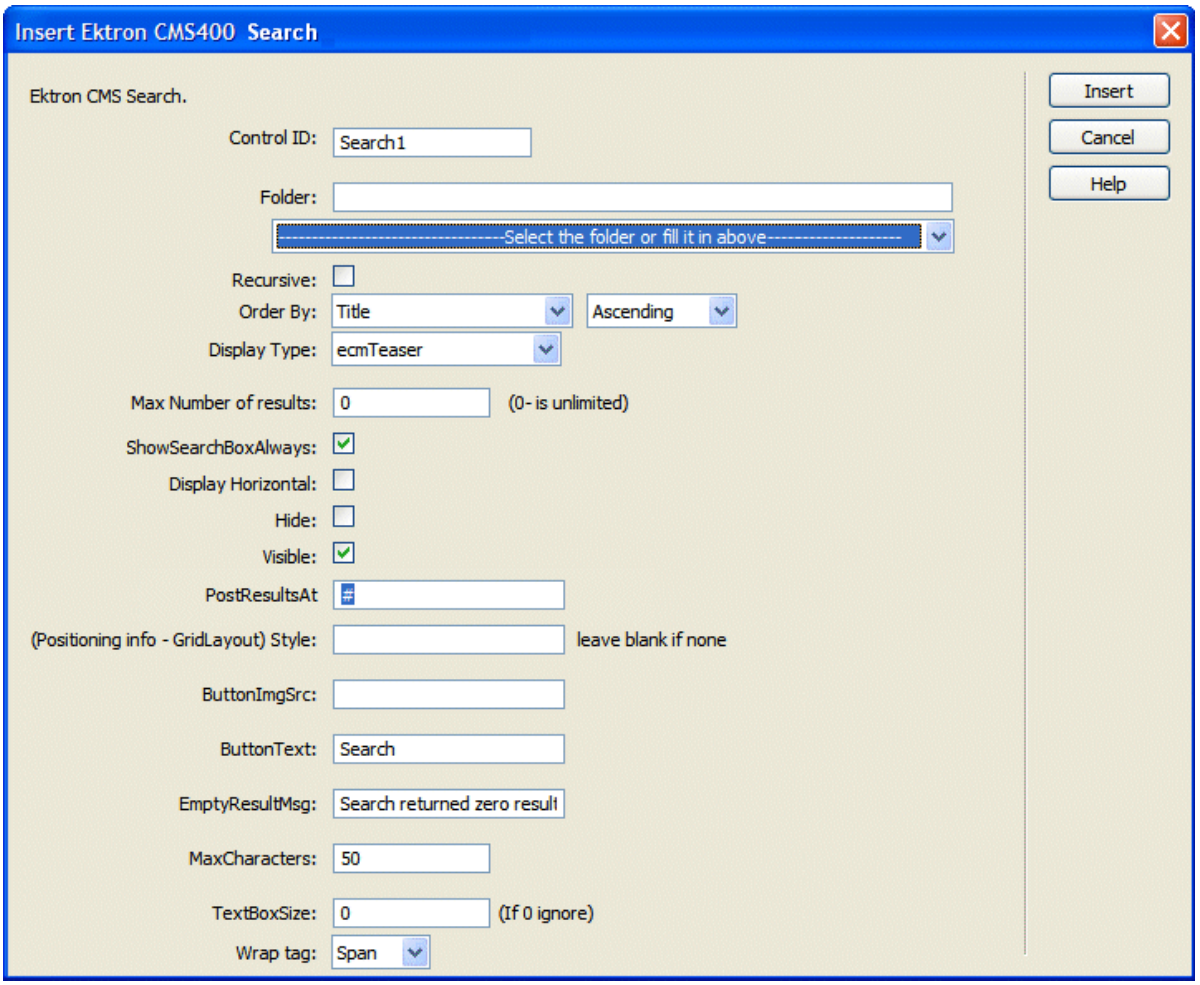
## Inserting a Search Function

For a description of the Search functions, see "Search Server Control" on page 85

To insert a Search function:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Content functions, click on **Search**

   The Insert Ektron Search box is displayed.



3. Use the following table to assist you with inserting a search.

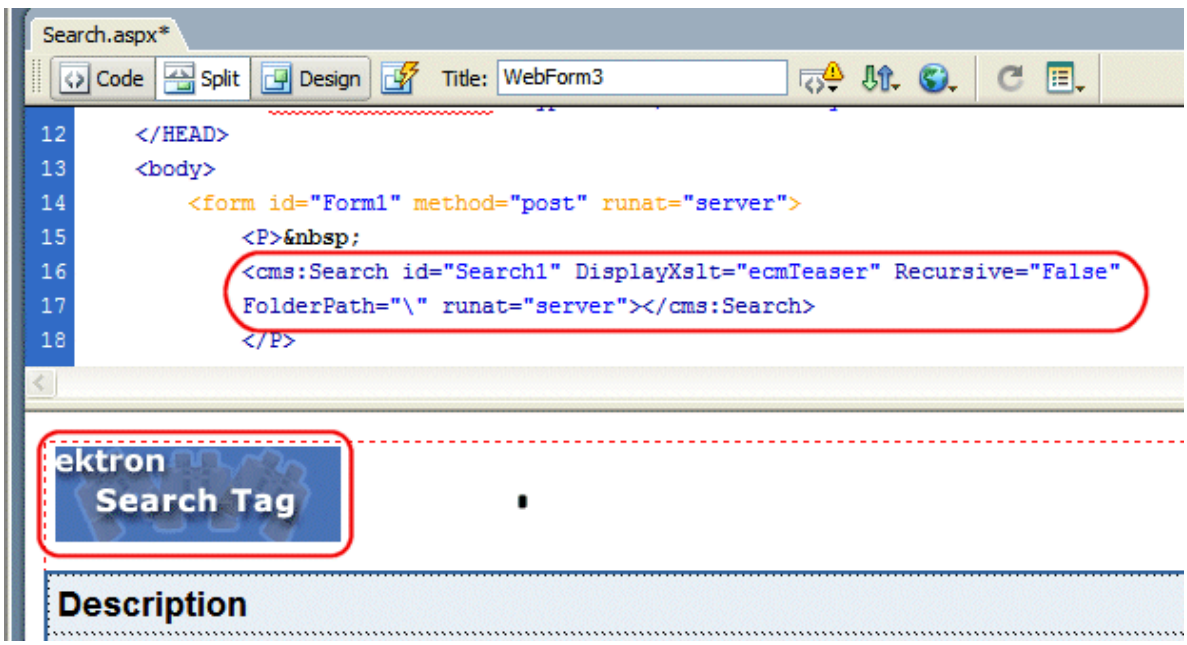
| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| Folder | Type in the folder path to be searched. If you do not know the folder path, you can navigate to it using the Select folder drop down box. |
| Select Folder | Select a folder path to be searched. If you insert the folder path, you do not need to select a folder. |
| Recursive | If checked, search information is gathered from the folder selected and all of its child folders. |
| Order By | Specify how you want to order the search results that are generated when a search is performed on you Web site. Also specify if you want the results in ascending or descending order.<br>• **Title** - The title of the content block<br>• **Date Created** - The date the content block was created<br>• **Date Modified** - The date the content block was last modified<br>• **Last Author Last Name** - The last author's last name<br>• **Last Author First Name** - The last author's first name<br>• **Ascending** - Alphabetic or numerical order<br>• **Descending** - Reverse alphabetic or numerical order |
| Display Type | Determines how the information on the page is displayed<br>**None**-databind only<br>**ecmNavigation** - lists the title of every content block in the folder<br>**ecmTeaser** - lists the title of every content block in the folder plus the content summary |
| Max Number of results | Enter a number to specify how may results are displayed on the page (0=unlimited). |
| ShowSearchBoxAlways | If **unchecked**, the search box does not appear on PostBack. |
| Display Horizontally | The search can be displayed either vertically or horizontally on your Web page. Check off this box if you want it to appear horizontally. |

| Field | Description |
|---|---|
| Hide | Used to hide a search in design time and run time.<br>**Checked** = Hide search<br>**Unchecked** = Show search<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| PostResultsAt | The path of the Web page where the search results will be posted.<br>The # symbol appears in the text box by default. The # symbol tells the control to post back to the Web page that created the search.<br>**Important:** A search control must exist on the posting page. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page.<br>For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |
| ButtonImgSrc | If you wish to use an image for the search button, insert the location of the image in this field. If no image is specified, a standard submit button is used. |
| ButtonText | The button text is the text that is displayed on the standard submit button, or the alt text of a search button image. |
| EmptyResultMsg | The return message displayed, if the search returns no hits.<br>The default is:<br>Search returned zero results |
| MaxCharacters | Specify the maximum amount of characters that is accepted in the text box. |
| TextBoxSize | Specify the size of the text box that is displayed on the Web page for users to enter the search keywords. The width is based on the number of characters. |

| Field | Description |
|---|---|
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |

4. When completed, click **insert**.

   A graphic defines where the search appears on your template.




Shown below is an example of the code that is inserted:

| Platform | Code View |
|---|---|
| .NET | `<cms:Search id="Search1" DisplayXslt="ecmTeaser" Recursive="False" FolderPath="\" runat="server"></cms:Search>` |

## Inserting an Event Calendar Function

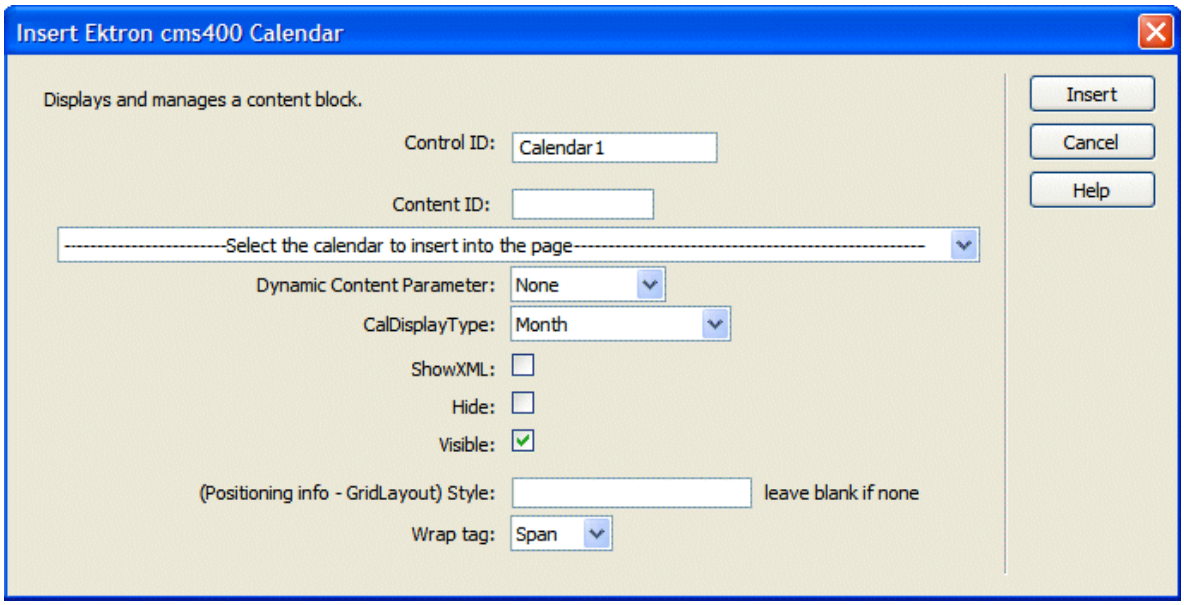
For a description of the Event Calendar function, see "Calendar Server Control" on page 127.

To insert an event calendar function:

**NOTE** Inserting a calendar function does not add a reference to a calendar style sheet. This step must be performed manually.

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click on **Calendar**

   The Insert Calendar box is displayed.



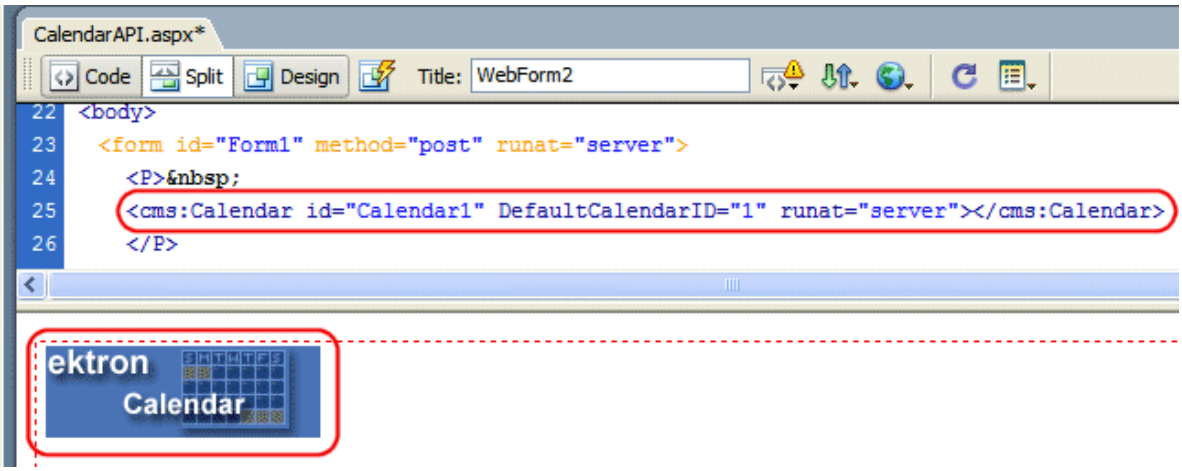

3. Use the following table to assist you with inserting a calendar.

| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| Content ID | Type in the Content ID of the calendar that you want to insert. If you do not know the Content ID, you can navigate to it using the Select Calendar drop down box. |
| Select Calendar | The drop-down box is populated with a list of all the calendar titles. When a title is selected, the content ID textbox is filled with that content block's id number. If you insert the calendar ID, you do not need to select a calendar. |
| Dynamic Content Parameter | To make this calendar dynamic, select **calendar_id**. When you do, this server control uses the calendar passed as a URL parameter. |

| Field | Description |
|---|---|
| CalDisplayType | Specify the Calendar view type. The default is **Month**.<br>• **Month** - A 30 day calendar month view.<br>• **Day** - Single day view.<br>• **Inline** - Display events that occur from the start date of the calendar through the end date.<br>• **MonthUpWithEvent** - Displays a month calendar. Events appear to the right of the calendar when you click on a day. |
| ShowXML | At runtime, output the raw XML data instead of the calendar. With this XML data, you can apply your own XSL to create your own custom display formats.<br>Default is **Unchecked.**<br>**Checked** = Show raw XML data<br>**Unchecked** = Show Calendar |
| Hide | Used to hide a calendar in design time and run time.<br>**Checked** = Hide calendar<br>**Unchecked** = Show calendar<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page.<br>For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |

4. When completed, click **insert**.

   A graphic defines where the calendar appears on your template.




Shown below is an example of the code that is inserted.

| Platform | Code View |
|---|---|
| .NET | `<cms:Calendar id="Calendar1" DefaultCalendarID="1" runat="server"></ cms:Calendar>` |

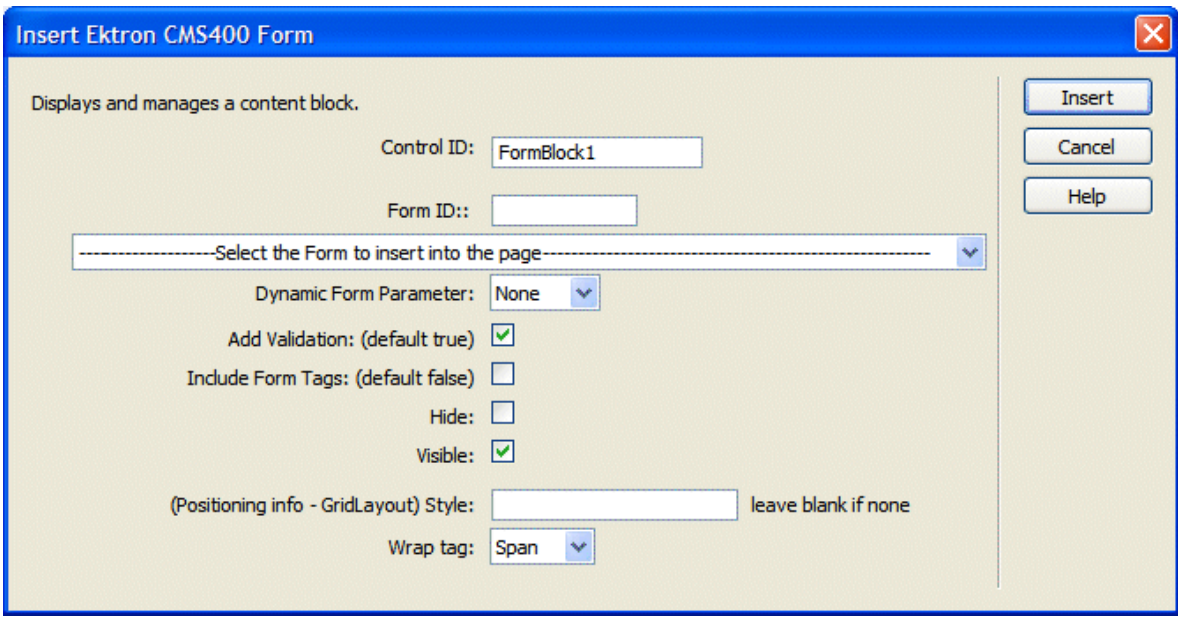
## Inserting a Form

For a description of the Form function, see ”FormBlock Server Control” on page 81

To insert a form:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Content functions, click **Form**

   The Insert Form box is displayed.



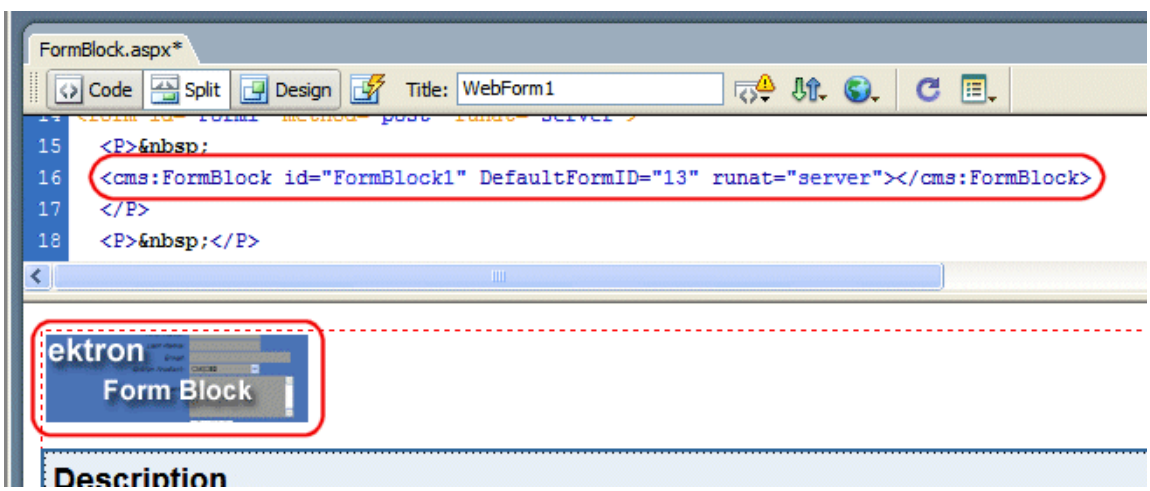

3. Use the following table to assist you with inserting a form.

| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| Form ID | Type in the form ID that you want to insert. If you do not know the Form ID, you can navigate to it using the Select Form drop down box. |
| Select Form | The drop-down box is populated with a list of all the form titles. When a title is selected, the form ID textbox is filled with that form's id number. If you insert the Form ID, you do not need to select a form. |
| Dynamic Form Parameter | To make this form block dynamic, select **ekfrm**. When you do, this control uses the form block passed as a URL parameter. |
| Add Validation | When **checked**, adds Validation to your main runat=server form. |
| Include Form Tags | Determines if tags are generated automatically or manually. *See Also:* "Automatic versus Manual Generation of Form Tags" on page 84 |

| Field | Description |
|---|---|
| Hide | Used to hide a form in design time and run time.<br>**Checked** = Hide form<br>**Unchecked** = Show form<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page.<br>For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |

4. When completed, click **insert**.

   A graphic defines where the form appears on your template.



Shown below is an example of the code that is inserted.

| Platform | Code View |
|---|---|
| .NET | `<cms:FormBlock id="FormBlock1" DefaultFormID="13" runat="server"></cms:FormBlock>` |

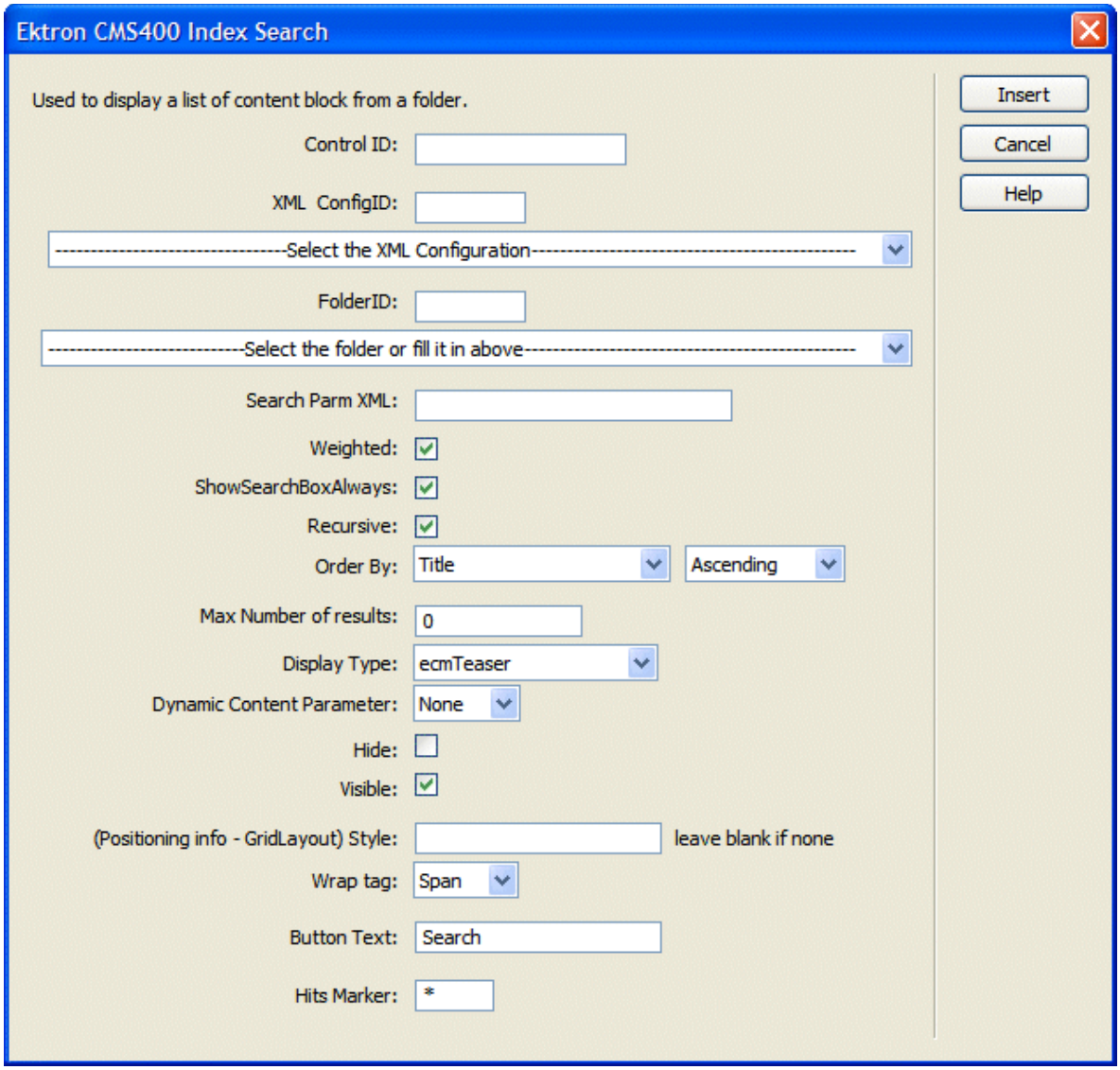
## Inserting an XML Index Search

For a description of the XML Index Search Function, see ”IndexSearch Server Control” on page 97.

To insert an XML Index Search:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET functions, click **XML Index Search**.

   The Insert XML Index Search box is displayed.

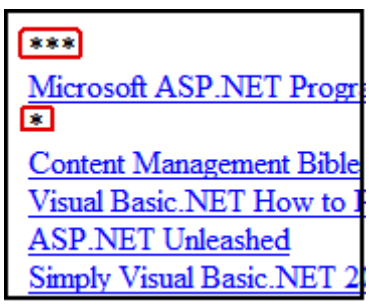



3. Use the following table to assist you with inserting an XML Index Search.

| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |

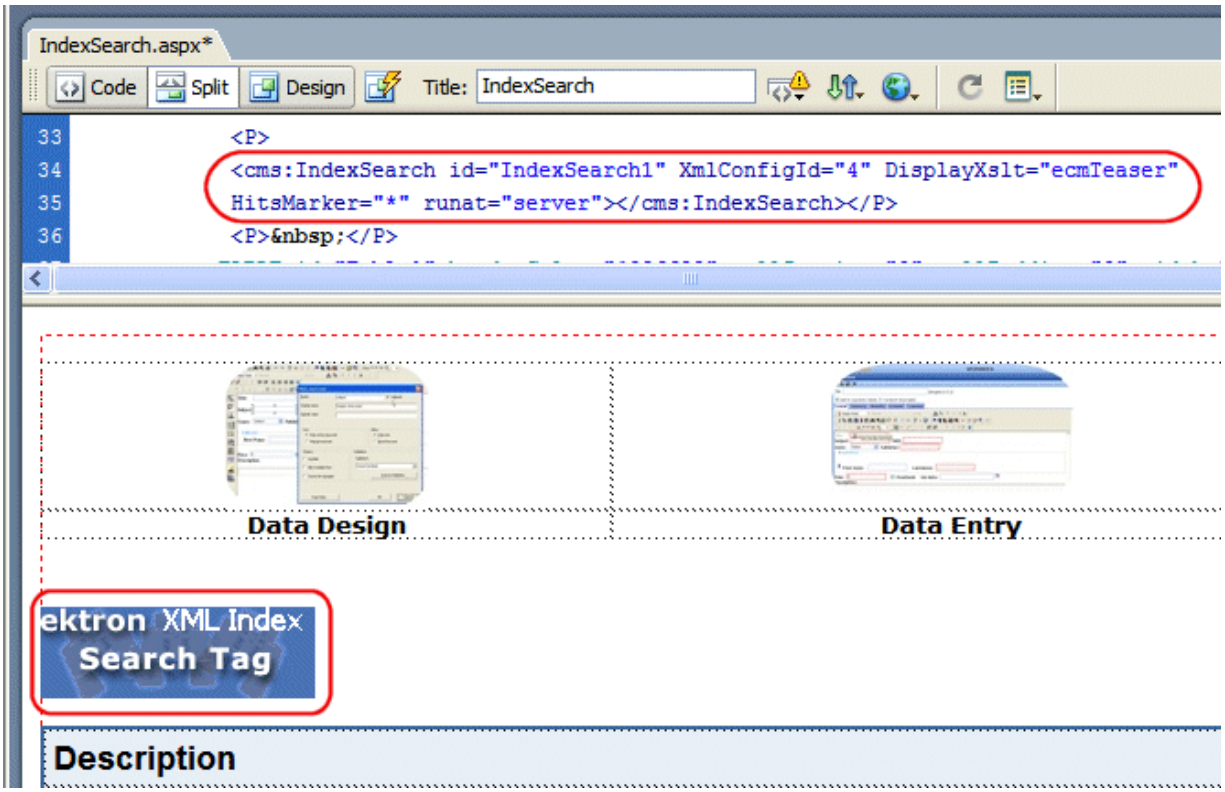
| Field | Description |
|---|---|
| XML ConfigID | ID of the XML configuration whose indexed fields appear on the search screen. If you do not know the XML ConfigID, you can navigate to it using the Select XML Configuration drop down box. |
| Select XML Configuration | The drop-down box is populated with a list of all the XML Configuration titles. When a title is selected, the XML ConfigID textbox is filled with that XML Configuration's id number. If you insert the XML ConfigID, you do not need to select a XML Configuration. |
| FolderID | The numeric id of the folder that you want to search. If you do not know the folder path, you can navigate to it using the Select folder drop down box. |
| Select Folder | Select a folder path to be searched. When a folder is selected, the folderID textbox is filled with that folder's id number. If you insert the Folder ID, you do not need to select a folder. |
| Search Parm XML | This property preserves settings that a developer would created in Visual Studio. |
| Weighted | If this value is **checked**, content blocks are ranked by the number of criteria matches.<br>For example, if a content block matches all search criteria, it appears at the top of the results page. A hit marker appears for each match.<br>If a second content block only matches one criterion, it also appears on the results page underneath the first content block. One asterisk appears above the page's title to indicate the single match.<br>**Checked**- display every content block that matches *at least one* search criteria<br>**Unchecked** - show only content blocks that match *all* search criteria |
| ShowSearchBoxAlways | If set to **unchecked**, the search box does not appear on when the results are posted to the Web page. |
| Recursive | Determines whether the XML Index Search Control uses this folder's child folders in the search. |

| Field | Description |
|---|---|
| Order By | Specify how you want to order the search results that are generated when a search is performed on you Web site. Also specify if you want the results in ascending or descending order.<br>• **Title** - The title of the content block<br>• **Date Created** - The date the content block was created<br>• **Date Modified** - The date the content block was last modified<br>• **Last Author Last Name** - The last author's last name<br>• **Last Author First Name** - The last author's first name<br>• **Ascending** - Alphabetic or numerical order<br>• **Descending** - Reverse alphabetic or numerical order<br>When you choose weighted search results, Ascending and Descending use how many criteria match your search. For example, if you have weighted results, and you choose Ascending, the results that match the most criteria appear first. |
| Max Number of results | The Maximum number of content blocks returned (0=unlimited). |
| Display Type | Determines how the information on the page is displayed<br>**None**-databind only<br>**ecmNavigation** - lists the title of every content block in the folder<br>*See Also:* "Example of ecmNavigation Display" on page 122<br>**ecmTeaser** - lists the title of every content block in the folder plus the content summary<br>*See Also:* "Example of ecmTeaser Display" on page 123 |
| Dynamic Content Parameter | Checks the QueryString for the content block ID value and replaces the search with content a when ID is specified. Leave blank to always display the search. By making content dynamic, when the content displayed on a Web page contains a link to another piece of content, you can display the second piece of content in the same Web page space.<br>**None** - Use Default<br>**ID** - The control uses the content block passed as a URL parameter. |
| Hide | Used to hide a form in design time and run time.<br>**Checked** = Hide form<br>**Unchecked** = Show form<br>Note: If Hide and Visible are both checked, the property is hidden. |

| Field | Description |
|---|---|
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page. For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |
| Button Text | The text that appears on the button. |
| Hits Marker | This symbol shows how many search criteria each hit matched.<br>The following example shows the first hit met three of the criteria. The second group met one criterion.<br>  |

4. When completed, click **insert**.

   A graphic defines where the XML Index Search appears on your template.




Shown below is an example of the code that is inserted.

| Platform | Code View |
|---|---|
| .NET | `<cms:IndexSearch id="IndexSearch1" XmlConfigId="4" DisplayXslt="ecmTeaser" HitsMarker="*" runat="server"></cms:IndexSearch>` |

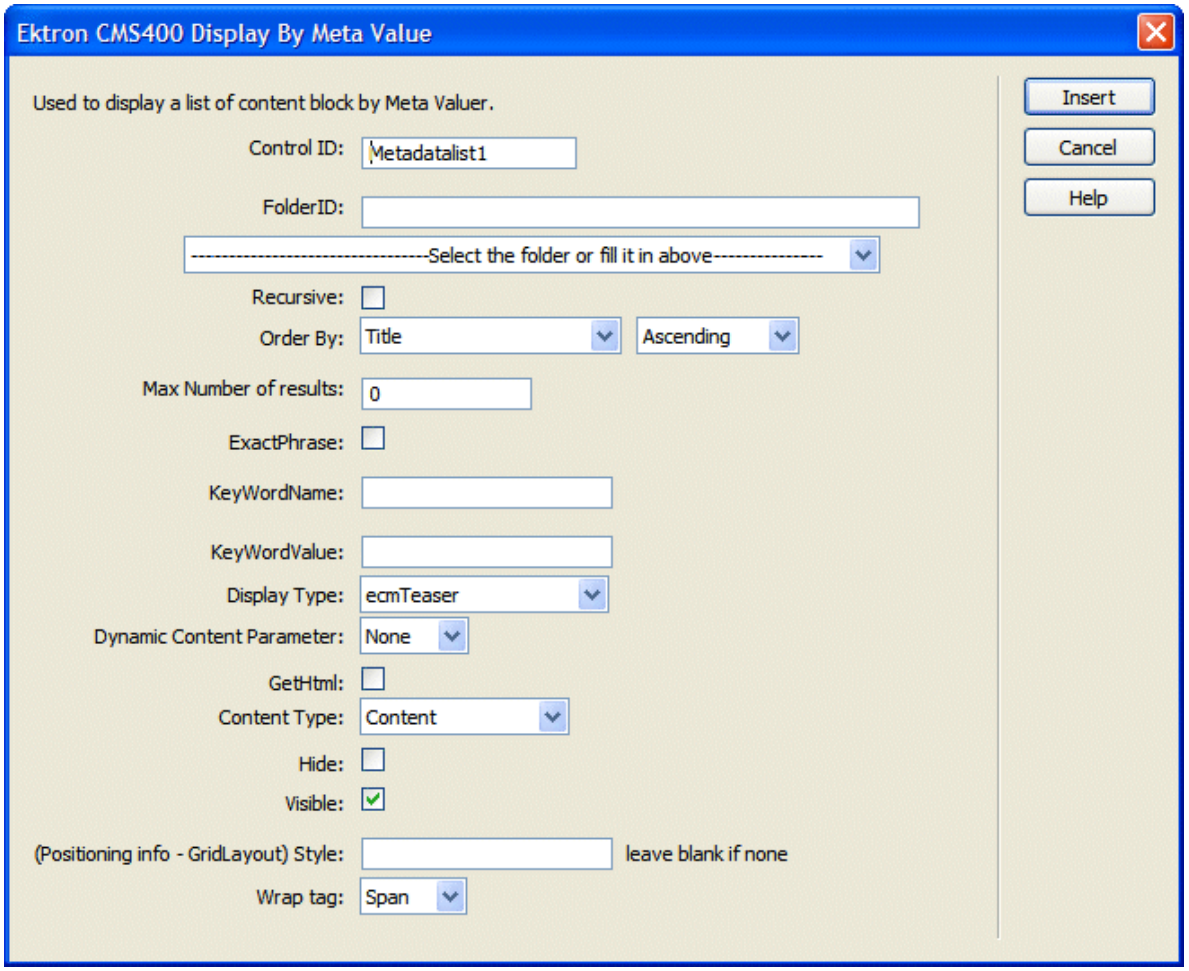
## Inserting Display By Meta Value

For a description of the Display By Meta Value function, see "MetaDataList Server Control" on page 189.

To insert Display By Meta Value:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Display By Meta Value**

   The Display By Meta Value box appears.




3. Use the following table to assist you with inserting Display By Meta Value.

| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| FolderID | The numeric id of the folder that you want to use. If you do not know the folder ID, you can navigate to it using the Select folder drop down box. |

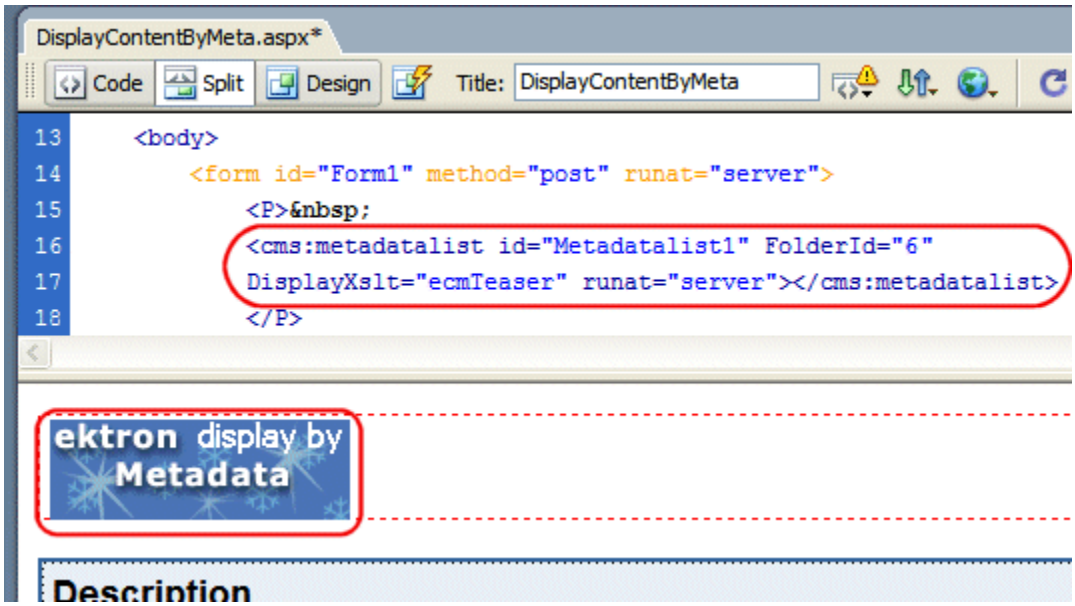
| Field | Description |
|---|---|
| Select Folder | Select a folder path to be used. When a folder is selected, the folderID textbox is filled with that folder's id number. If you insert the Folder ID, you do not need to select a folder. |
| Recursive | Determines whether the Display By Meta Value control uses this folder's child folders. |
| Order By | • Specify how you want to order the results that are generated. Also specify if you want the results in ascending or descending order.<br>• **Title** - The title of the content block<br>• **Date Created** - The date the content block was created<br>• **Date Modified** - The date the content block was last modified<br>• **Last Author Last Name** - The last author's last name<br>• **Last Author First Name** - The last author's first name<br>• **ID** - The content blocks ID<br>• **Ascending** - Alphabetic or numerical order<br>• **Descending** - Reverse alphabetic or numerical order |
| Max Number of results | The Maximum number of content blocks returned (0=unlimited). |
| ExactPhrase | Determines whether the KeyWordValue needs to match the phrase exactly.<br>For example, if "site" is the KeyWordValue and the title of a content block is "Welcome to the site" and ExactPhrase is set to true, you would not see the content block in the metadata list. This is because "site" does not equal "Welcome to the site".<br>**Checked** = Match the exact phrase<br>**Unchecked** = Doesn't need to match exact phrase |
| KeyWordName | The KeyWordName is the container where the KeyWordValues are located.<br>Some examples of a KeyWordName might be "Keywords or Title."<br>For information on creating KeyWordNames for Metadata, see the **Ektron CMS400.NET** Administrators Manual > "Managing Content" > "Metadata" > "Managing Metadata Definitions" > "Adding a Metadata Definitions" |
| KeyWordValue | The value that is associated with the KeyWordName.<br>Some examples of a KeyWordValue might be "Home; page; or company" |

| Field | Description |
|---|---|
| Display Type | Determines how the information on the page is displayed.<br>**None**-databind only<br>**ecmNavigation** - lists the title of every content block in the folder<br>*See Also:* "Example of ecmNavigation Display" on page 122<br>**ecmTeaser** - lists the title of every content block in the folder plus the content summary<br>*See Also:* "Example of ecmTeaser Display" on page 123 |
| Dynamic Content Parameter | Checks the QueryString for the content block ID value and replaces the Display By Meta Value Control with content when an ID is specified. Leave blank to always display the Display By Meta Value Control.<br>**None** - Use Default<br>**ID** - The control uses the content block passed as a URL parameter. |
| GetHtml | Set to **True** if you want to retrieve and display content (html body) for all content blocks in the collection. For example, display content inside a web server control such as a DataGrid. |
| Content Type | Defines what type of list summary appears.<br>**Content** - Displays only content blocks in the folder.<br>**All Types** - Displays all content in the folder.<br>**Forms** - Displays only forms in the folder.<br>**Archive_Content** - Displays archived content from the folder.<br>**Archive_Forms** - Displays archived forms from the folder. |
| Hide | Used to hide the function in design time and run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page. For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |

| Field | Description |
|---|---|
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |

4. When completed, click **insert**.

   A graphic defines where the Display By Meta Value appears on your template.




Shown below is an example of the code that is inserted:

| Platform | Code View |
|---|---|
| .NET | `<cms:metadatalist id="Metadatalist1" FolderId="69" DisplayXslt="ecmTeaser" runat="server"></cms:metadatalist>` |

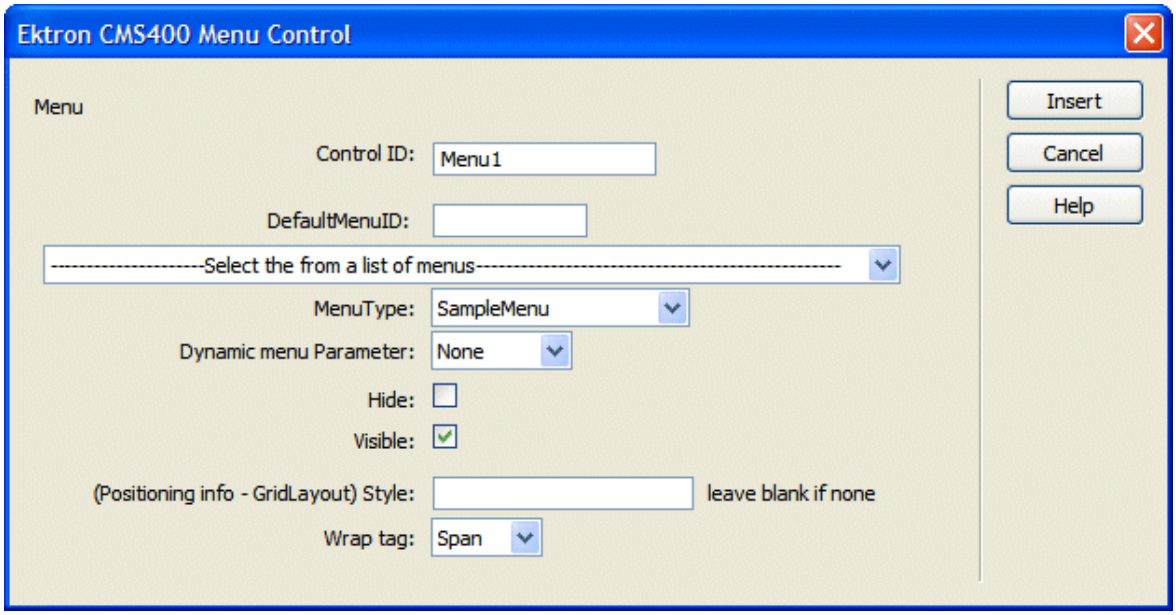
## Inserting a Standard Menu

For a description of the Standard Menu function, see "Menu Server Control" on page 161.

To insert a Standard Menu:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Standard Menu**

   The Standard Menu Control box appears.



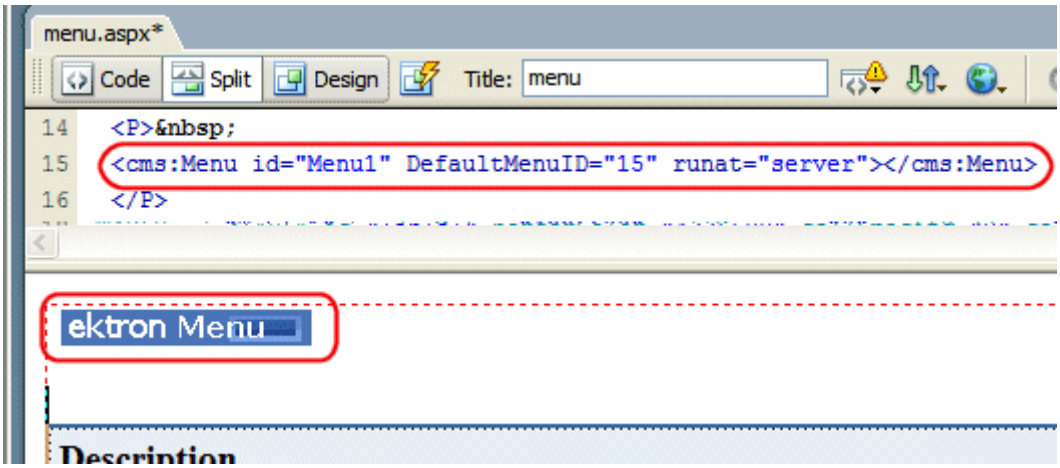

3. Use the following table to assist you with inserting a Standard Menu.

| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| DefaultMenuID | The numeric id of the menu that you want to insert. If you do not know the DefaultMenuID, you can navigate to it using the Select Menu drop down box. |
| Select Menu | Select a menu to be used. When a menu is selected, the DefaultMenuID textbox is filled with that menu's id number. If you insert the DefaultMenuID, you do not need to select a menu. |
| MenuType | The type of menu to render.<br>**None** - databind only<br>**SampleMenu** - A sample display, formatted as a bulleted menu list<br>**TreeMenu** - A sample display, formatted as a folder tree. You can expand the tree by clicking on the folder icon.<br>For more information on using the SampleMenu and TreeMenu Display, see "Using DisplayXslt Samples" on page 164. |

| Field | Description |
|---|---|
| Dynamic Menu Parameter | Checks the QueryString for the menu value and replaces the menu with content when ID is specified. Leave blank to always display the menu.<br>**None** - Use Default<br>**menu_ID** - The control uses the menu passed as a URL parameter. |
| Hide | Used to hide a menu in design time and run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page.<br>For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |

4. When completed, click **insert**.

   A graphic defines where the Standard Menu appears on your template.




Shown below is an example of the code that is inserted:

| Platform | Code View |
|---|---|
| .NET | `<cms:Menu id="Menu1" DefaultMenuID="15" runat="server"></cms:Menu>` |

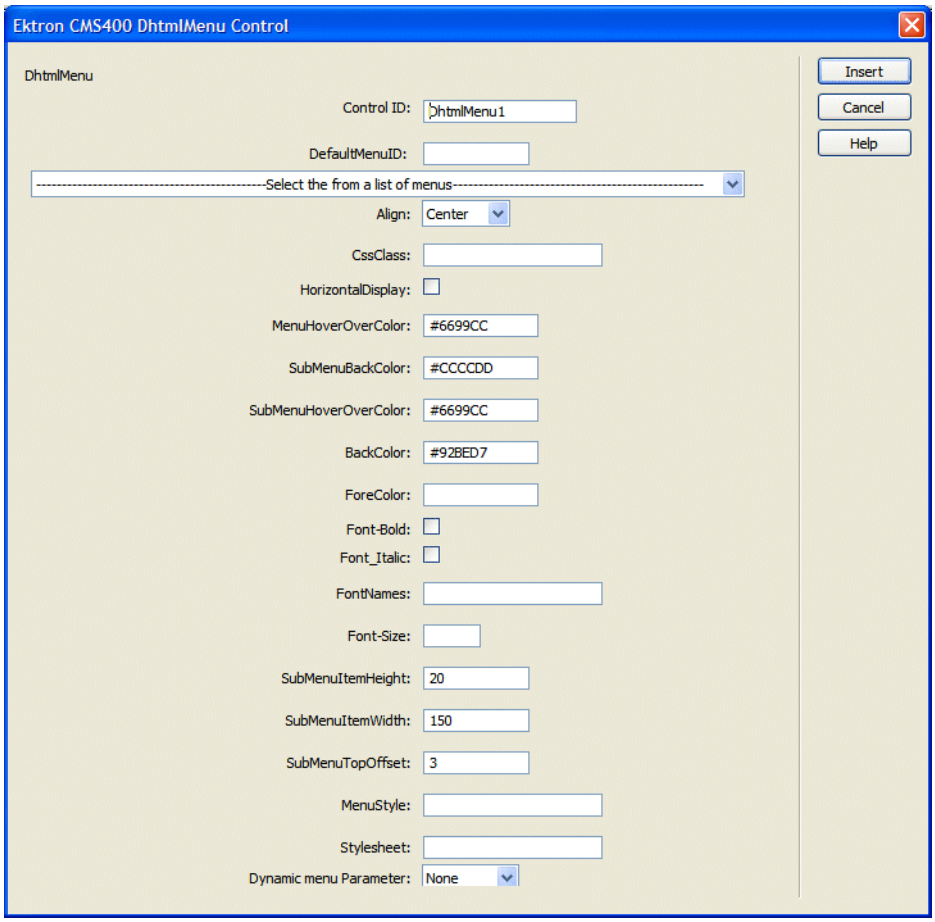
## Inserting a DHTML Menu

For a description of the DHTML Menu function, see "DhtmlMenu Server Control" on page 178.

To insert a DHTML Menu:

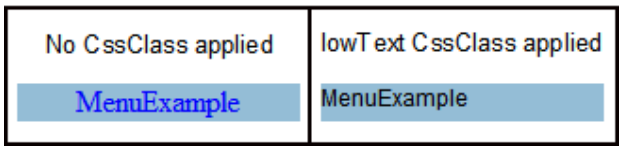
1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **DHTML Menu**

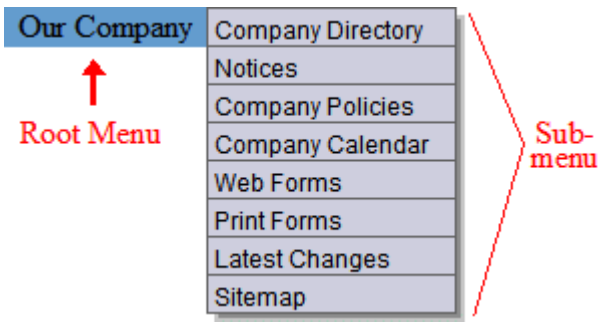
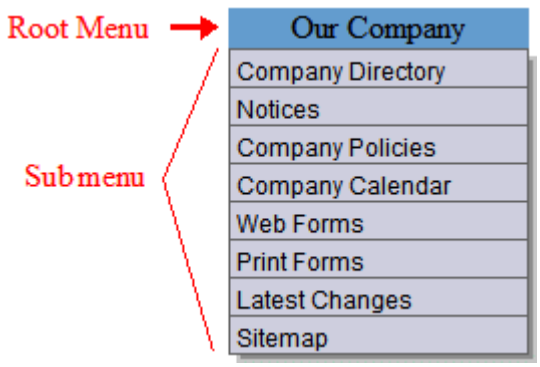
   The DHTML Menu Control box appears.

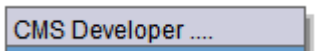
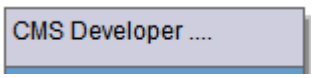
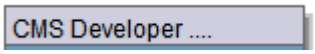
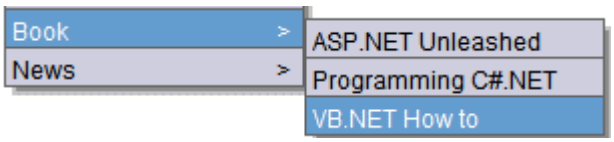
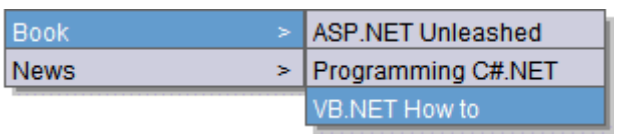
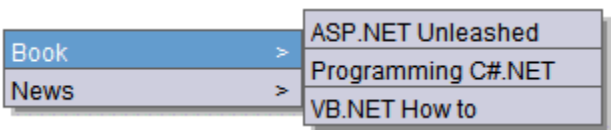



3. Use the following table to assist you with inserting a DHTML Menu.

| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| DefaultMenuID | The numeric id of the menu that you want to insert. If you do not know the DefaultMenuID path, you can navigate to it using the Select Menu drop down box. |

| Field | Description |
|---|---|
| Select Menu | Select a menu to be used. When a menu is selected, the DefaultMenuID textbox is filled with that menu's id number. If you insert the DefaultMenuID, you do not need to select a menu. |
| Align | Specifies the horizontal alignment of the root menu text.<br>The default is **Center.**<br>**NotSet** - Use the default setting.<br>**Left** - Aligns text to the left.<br>**Center** - Aligns text to the center.<br>**Right** - Aligns text to the right.<br>**Justify** - Aligns text justified to both margins. |
| CssClass | Apply a .css class to the root menu. Enter the class name from the .css defined in the Style sheet property.<br>Since pop_style.css is the default style sheet, you can use it's classes without defining it in the Stylesheet property.<br>For example enter lowtext in the property, the root menu changes as shown below.<br>  |

| Field | Description |
|---|---|
| HorizontalDisplay | Displays submenu to the right of the root menu. This only affects the first submenu of the root menu. All subsequent submenus appear to the right of each submenu.<br>The default is **False**<br>**True** = Submenu appears to the right of the root menu<br>  <br><br>**False** = Submenu appears below the root menu<br>  <br> |
| MenuHoverOverColor | Sets the background color for the root menu when hovering over it. |
| SubMenuBackColor | Sets the background color for the submenu. |
| SubMenuHoverOverColor | Sets the background color for the submenu when hovering over it. |
| BackColor | Sets the background color for the root menu. This property appears in the **Appearance** category |
| ForeColor | Sets the color of the font for the root menu. This property appears in the **Appearance** category. |

| Field | Description |
|---|---|
| Font-Bold | Checking this box allows the font to appear **bold**. |
| Font_Italic | Checking this box allows the font to appear *italics.* |
| FontNames | Enter the name of a font to use. |
| Font-Size | The size of the font. |
| SubMenuItemHeight | Sets the height of each submenu item in pixels. The Default is 20<br>Example set to 20:  <br>Example set to 30:  |
| SubMenuItemWidth | Sets the width of each submenu item in pixels. The Default is 150<br>Example set to 150:  <br>Example set to 130: |
| SubMenuTopOffset | Sets the location of the top of the submenu. This number is in pixels. The default is 5<br>Example set to 5:<br>  <br>Example set to 0 (zero):<br>  <br>Example set to -10:<br>  |

| Field | Description |
|---|---|
| MenuStyle | A specified inline style that defines the root menu.<br>For example enter the following in the MenuStyle property box:<br>`font: bold 18px Arial;color:royalblue; background-color:slateblue`<br> |
| StyleSheet | Specify the location and style sheet for the submenu. For example: `\Workarea\csslib\mytest.css.`<br>Leave blank to use the default.<br>`pop_style.css` is the default style sheet. It is located in `\webroot\CMS400Example\Workarea\csslib`. For more information on style sheets, see "Using a Style Sheet" on page 183<br>The Stylesheet property can be used in conjunction with the CssClass property to define the root menu. |
| Dynamic Menu Parameter | Checks the QueryString for the menu value and replaces the menu with content when ID is specified. Leave blank to always display the menu.<br>**None** - Use Default<br>**menu_ID** - The control uses the menu passed as a URL parameter. |

4. When completed, click **insert**.

   A graphic defines where the DHTML Menu appears on your template.

   Shown below is an example of the code that is inserted:

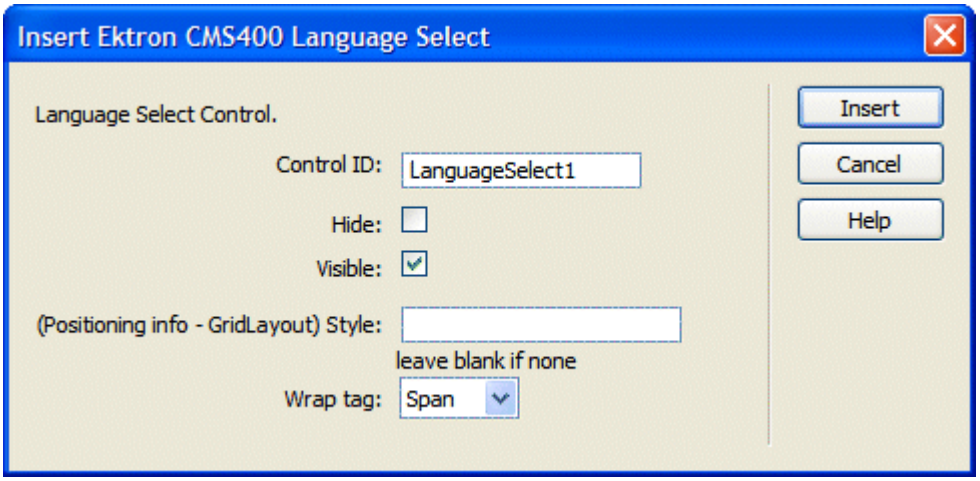
| Platform | Code View |
|---|---|
| .NET | `<cms:DhtmlMenu id="DhtmlMenu1" DefaultMenuID="16" ForeColor="" runat="server"></cms:DhtmlMenu>` |

## Inserting a Language Select Box

For a description of the Language Select Box, see "LanguageSelect Server Control" on page 65.

To insert a Language Select Box:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Language Select**
3. The Language Select box appears.



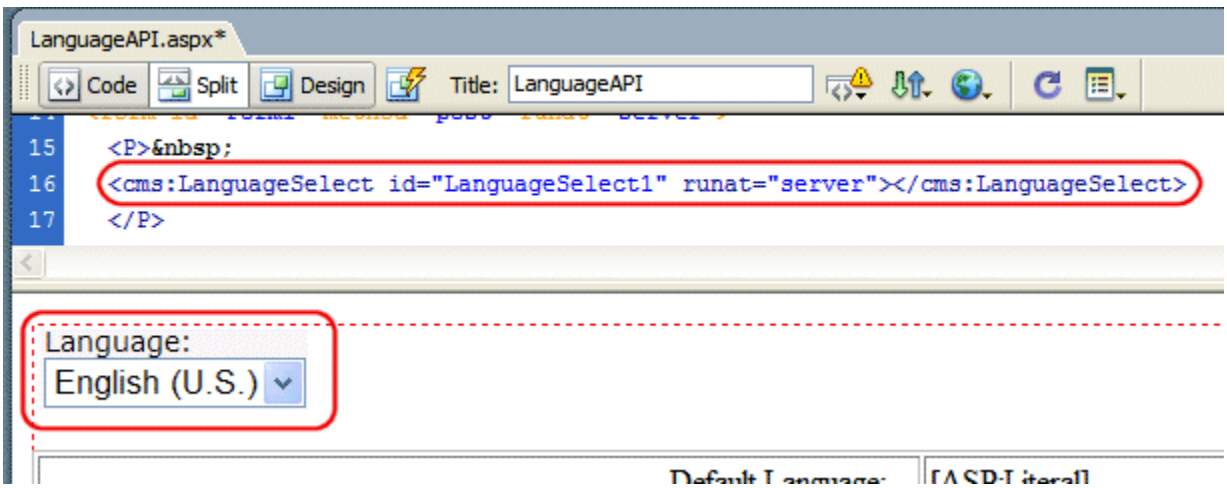

4. Use the following table to assist you with inserting a Language Select box.

| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| Hide | Used to hide a function in design time and run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page.<br>For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |

| Field | Description |
|---|---|
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |

5. When completed, click **insert**.

   A graphic defines where the Language Select appears on your template.




Shown below is an example of the code that is inserted.

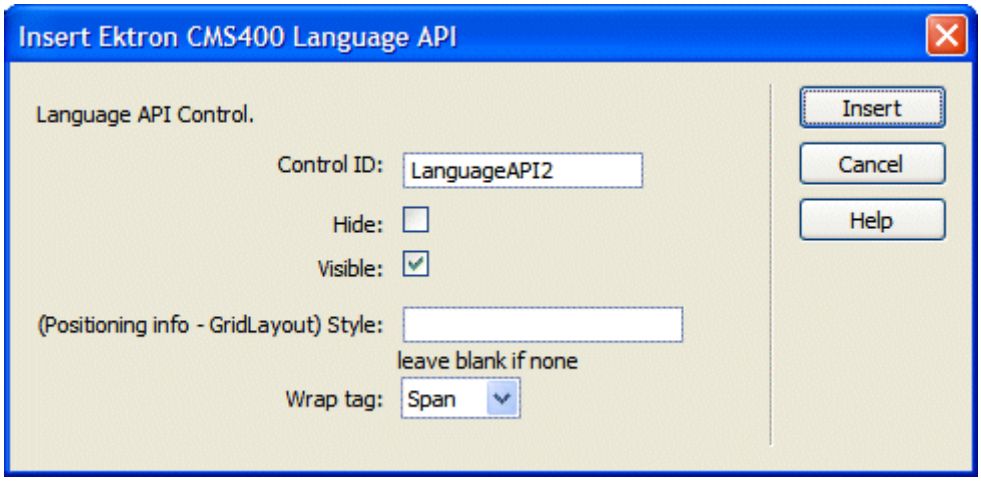
| Platform | Code View |
|---|---|
| .NET | `<cms:LanguageSelect id="LanguageSelect1" runat="server"></ cms:LanguageSelect>` |

## Inserting a Language API

For a description of the Language API function, see "LanguageAPI Server Control" on page 68.

To insert a Language API:

1. Move the blinking cursor to the area that you want to place it.
2. From the list of Ektron CMS400.NET Functions, click **Language API**
3. The Language API box appears.



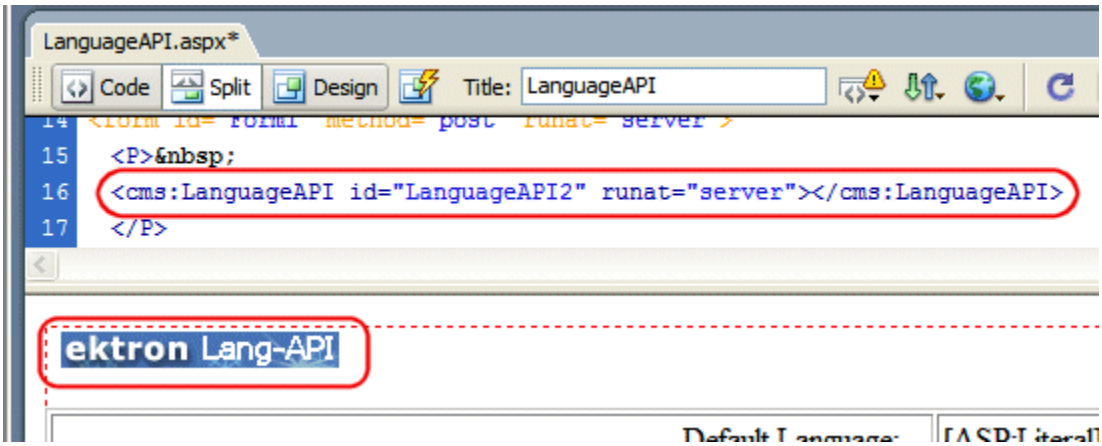

4. Use the following table to assist you with inserting a Language API box.

| Field | Description |
|---|---|
| Control ID | A unique name for the function you are inserting. |
| Hide | Used to hide a Language API in design time and run time.<br>**Checked** = Hide Language API<br>**Unchecked** = Show Language API<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Visible | Used to show or hide the function at run time.<br>**Checked** = Hide function<br>**Unchecked** = Show function<br>Note: If Hide and Visible are both checked, the property is hidden. |
| Style (Positioning Info - GridLayout) | Set position information of a control as absolute, when using a GridLayout Web page.<br>For example:<br>style="Z-INDEX: 101; LEFT: 328px; POSITION: absolute; TOP: 240px" |

| Field | Description |
|---|---|
| Wrap Tag | Allows a developer to specify a server control's tag.<br>The default is **Span**.<br>**Span** - The  tag is used to designate an in-line portion of an HTML document as a span element.<br>**Div** - The  tag is used when you want to apply attributes to a block of code. |

5. When completed, click **insert**.

   A graphic defines where the Language API appears on your template.




Shown below is an example of the code that is inserted:

| Platform | Code View |
|---|---|
| .NET | `<cms:LanguageAPI id="LanguageAPI2" runat="server"></cms:LanguageAPI>` |

# Standard ASP.NET Methods

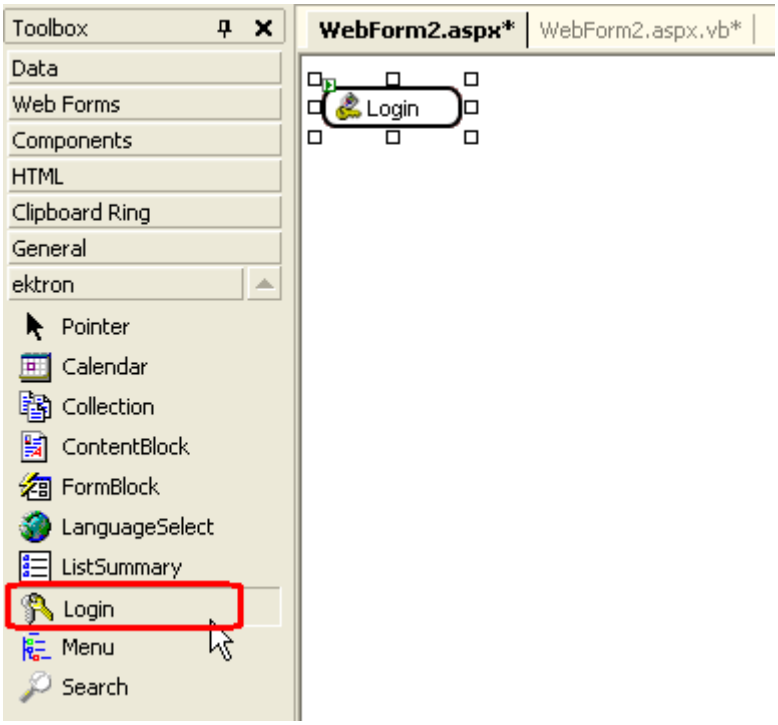
Use these methods to customize **Ektron CMS400.NET**. To use them, insert them onto your template pages to invoke content blocks, list summaries, and other objects stored in your database. They are the primary building blocks of your **Ektron CMS400.NET** Content Management System.

## Replacement of Standard Methods

Beginning with Release 4.7, **Ektron CMS400.NET** utilizes ASP.NET server controls, which replace most Standard ASP.NET methods. With **Ektron CMS400.NET**'s new toolbox of pre-built, reusable, data-bindable ASP.NET controls, Ektron empowers developers working in Visual Studio .NET to rapidly integrate Ektron CMS components into their templates and sites. For more Information on Ektron CMS400.NET controls, see "Introduction to Ektron CMS400.NET Server Controls" on page 24.

The following examples shows the Login Server Control added by drag and drop vs. adding a Login method by the ASP.NET method.

Drag and Drop Login Server Control




Adding a Login function via an ASP.NET method.

```
<tr>

    <td>

        <%=AppUI.ecmLogin()%>

    </td>

</tr>
```

**Best Practice**

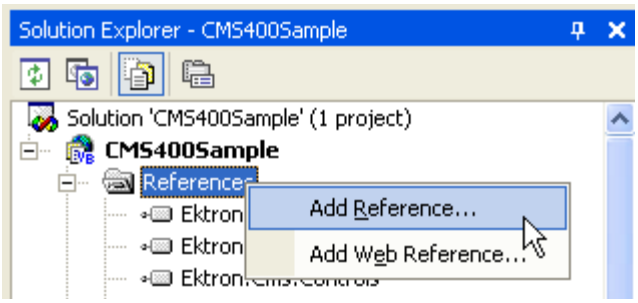
Ektron recommends using CMS400.NET server controls when available over Ektron standard ASP.NET methods. The methods are being replaced by the server controls.

**NOTE** You can still use the standard ASP.NET methods in **Ektron CMS400.NET**.

# Referencing Methods in Visual Basic

Before you can use methods with Visual Basic, you must add a reference to them from Visual Basic. Follow these steps to do so.

1. Open Visual Basic.
2. Choose or add a project.
3. In Solution Explorer, right click **References** and choose **Add Reference**.

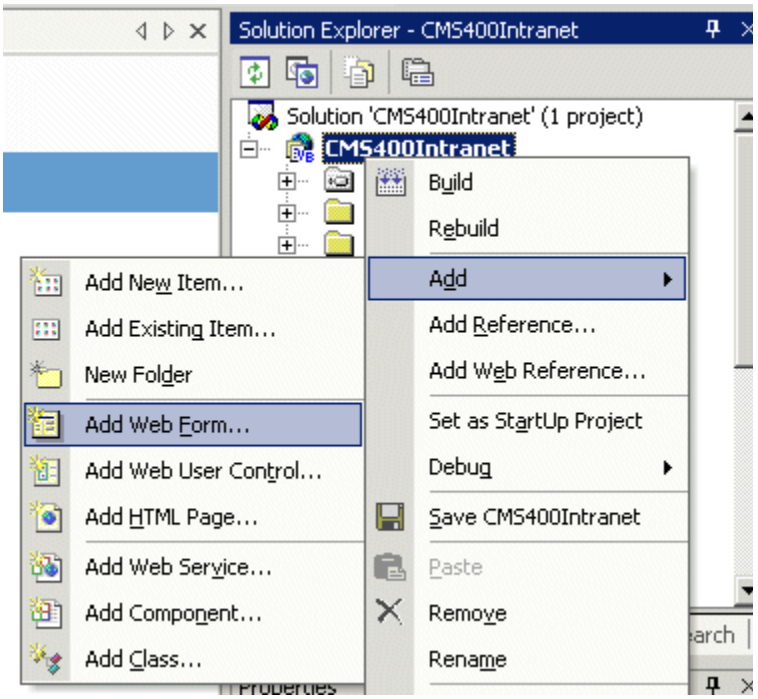



4. In the Add Reference dialog box, click **Browse**.
5. Go to `Inetpub/wwwroot/CMS400Example/Bin`.
6. To load the standard methods click `Ektron.Cms.UI.CommonUI.dll`. To load the XML Indexing methods, click `Ektron.Cms.Common.dll`.
7. Click **Open**, then **OK**.

# Adding a New Web Form

To add a new Web form, follow these steps.

1. Open your project in Visual Studio .NET.
2. Right click the project.
3. Click **Add > Add Web Form**.




4. Click **Web Form**.
5. In the code behind, add an `Imports Ektron.Cms.UI.CommonUI` statement.

# Requirements for Standard Methods

Ektron CMS400.NET maintains and displays content through the use of Ektron ASPX methods and ASP.NET assembly objects. To use the methods, include the following code in your template:

```
<%@ Page Language="vb" AutoEventWireup="false" %>

<%@ Import Namespace="Ektron.Cms.UI.CommonUI" %>

<% Dim AppUI As New ApplicationAPI %>
```

Each line is explained below.

| Line | Description |
|---|---|
| `<%@ Page Language="vb" AutoEventWireup="false" %>` | A generic .net line of code that should always be placed in the template |
| `<%@ Import Namespace="Ektron.Cms.UI.CommonUI" %>` | An import statement that calls the namespace `Ektron.CMS.UI.CommonUI` to display information from the .net assembly file |
| `<% Dim AppUI As New ApplicationAPI %>` | Creates an instance of the ApplicationUI class that can be used by each method |

These lines call the Ektron .net assembly object to allow the page to function properly. They must appear at the top of each template file.

# Standard Methods

To customize your Ektron CMS400.NET application, you need to be familiar with these methods.

| Method | Description | More Information |
|---|---|---|
| Add Content | Programmatically adds a content block(s) to Ektron CMS400.NET | ”Add Content” on page 268 |
| Collection | Displays links generated by collections. | ”Collection Server Control” on page 120 |
| Content Block | Displays and manages a content block. | ”ContentBlock Server Control” on page 71 |
| Event Calendar | Displays an event calendar | ”Calendar Server Control” on page 127 |
| Form | Displays a content block that is associated with a form. | ”FormBlock Server Control” on page 81 |

| Method | Description | More Information |
|---|---|---|
| IndexSearch | Displays on a Web page a set of search fields. A site visitor uses the page to search for information on your Web site. | "IndexSearch Server Control" on page 97 |
| List Summary | Displays a list of content block summaries. | "ListSummary Server Control" on page 114 |
| Login | Displays the login or logout button, which lets the user log in and use Ektron CMS400.NET. | "Login Server Control" on page 63 |
| Metadata | Displays the metadata in the source of the Web page displaying the content. | "MetaData Server Control" on page 186 |
| Menu | Customizes the DHTML menus feature | "DhtmlMenu Server Control" on page 178 |
| Search | Inserts a text box with a search button in the template. When the user completes the search screen, it looks through XHTML content on the Web site. | "Search Server Control" on page 85 |
| ShowSiteLanguage | Lets a site visitor select a language in which to view site. | "LanguageSelect Server Control" on page 65 |

# Add Content

The add content method lets you programmatically add a new content block to Ektron CMS400.NET without logging in. This method can be used for several applications, such as

- A Web form allowing you to collect data from users about the content block
- A loop function that moves information from a database to an Ektron CMS400.NET database

Here is the format for the add content method.

```
<% AddContent
(    UserName,
Password,
Domain,
content_title,
content_comment,
ContentHtml,
SummaryHTML,
MetaInfo,
ContentLanguage,
FolderID,
GoLive,
EndDate,
ErrString
)%>
```

The table below lists attributes of the add content block method.

| Attribute | Description | Type |
|---|---|---|
| UserName | The user to be authenticated for permissions for eligibility to add a content block. | As String |
| Password | The password for the username being authenticated. | As String |
| Domain | If using Active Directory, specify the domain needed to authenticate the user. | As String |
| Content_title | Title of content block being added. *See Also:* "Managing Content" in the **Ektron CMS400.NET** Administrator Manual | As String |
| Content_comment | Historical comment for content being added. | As String |
| ContentHTML | Content for content block. | As String |
| SummaryHTML | Summary information about content block. | As String |
| MetaInfo | Information for the metadata that will be used as a reference point for finding other words or information. | As Object |

| Attribute | Description | Type |
|---|---|---|
| ContentLanguage | Language of the content. *See Also:* "Multi-Language Support" in the **Ektron CMS400.NET** Administrator Manual | As String |
| FolderID | ID number of content folder to which content block will be added. *See Also:* "Managing Content Folders" in the **Ektron CMS400.NET** Administrator Manual | As String |
| GoLive | Date and time content will go live on the Web site.<br>Must match date/time format set in web.config. | As String |
| EndDate | Date and time content will no longer display on Web site.<br>Must match date/time format set in web.config. | As String |
| ErrString | String returned if content fails to get added to Ektron CMS400.NET. | As Object |

## Add Content Example

The following is an example of the add content method hard coded in a template.

```
<tr>

  <td>

    <% AddContent ("admin", "admin", "", "Content Title", "Comment",
    "This is the <b>body</b> of the content.", "This is the
    <i>summary</i> for the content.", 1, 31, "06-14-2004 11:55 AM",
    "", "<metadata><meta id=""3"">ContentTitle</meta></metadata>",

    "") %>

  </td>

</tr>
```

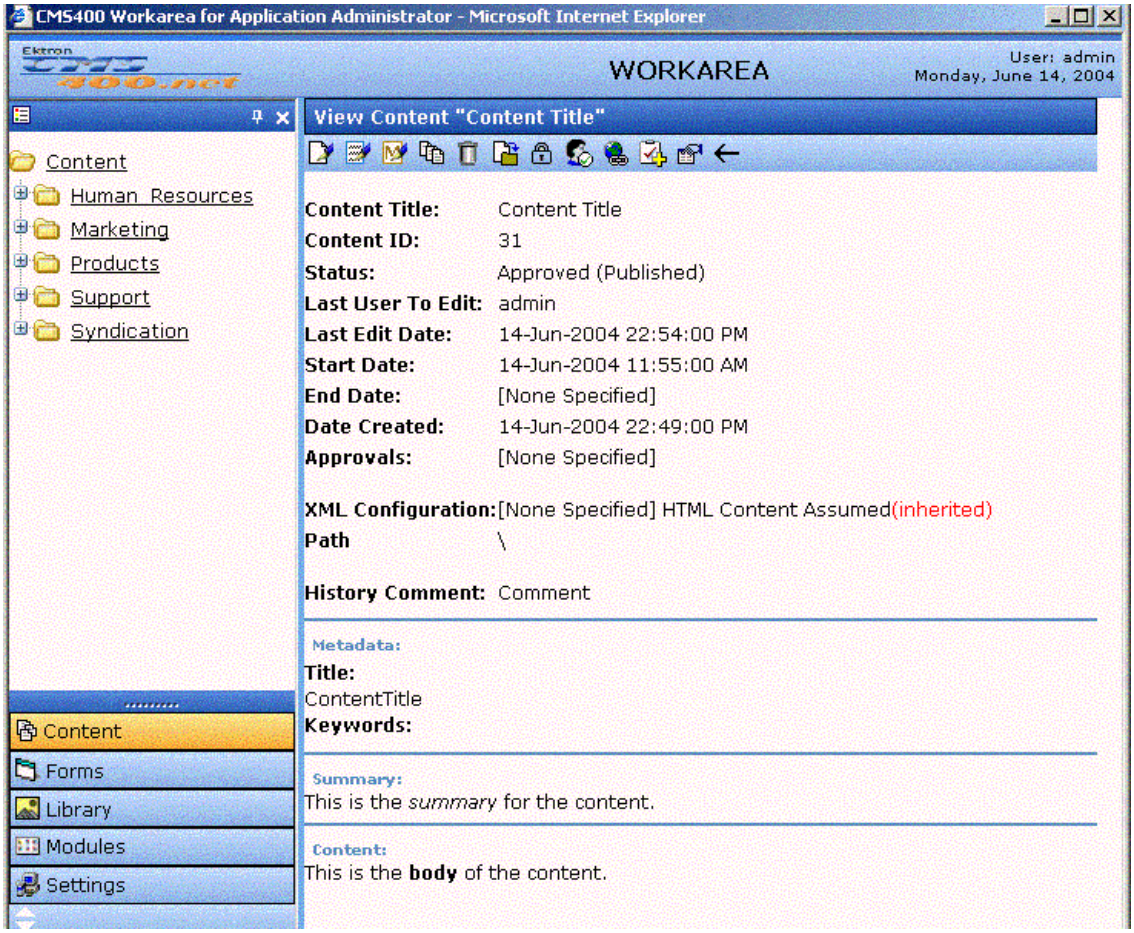
The previous example has the following properties:

- The user admin is used for authentication
- The title of the new content block is "Content Title"

- The comment for the content is “Comment”
- The body of the content is “This is the **body** of the content”
- The summary of the content block is “This is the *summary* for the content
- The language ID = 1033, or English
- The new content block is added to content folder ID=31
- The start date is set at June 14, 2004 at 11:55 AM
- No end date is specified
- The metadata “Content title” is set for the metadata definition ID=3

When this method is executed, the content is added to Ektron CMS400.NET. Displayed here is the record of the content in the workarea.

.


CMS400 Workarea for Application Administrator - Microsoft Internet Explorer
WORKAREA
User: admin
Monday, June 14, 2004
View Content "Content Title"
Content
Human Resources
Marketing
Products
Support
Syndication
Content
Forms
Library
Modules
Settings
Content Title: Content Title
Content ID: 31
Status: Approved (Published)
Last User To Edit: admin
Last Edit Date: 14-Jun-2004 22:54:00 PM
Start Date: 14-Jun-2004 11:55:00 AM
End Date: [None Specified]
Date Created: 14-Jun-2004 22:49:00 PM
Approvals: [None Specified]
XML Configuration: [None Specified] HTML Content Assumed(inherited)
Path \
History Comment: Comment
Metadata:
Title:
ContentTitle
Keywords:
Summary:
This is the summary for the content.
Content:
This is the body of the content.

# Custom ASP Functions

Ektron CMS400.NET displays content on your Web site through the use of custom Ektron ASP functions and ComObjects. Your template you must include the following file to be able to use the custom functions:

```
<!-- #include file="applicationAPI.asp" -->
```

This file (**`application.asp)`** calls the file with the custom Ektron functions to allow the page to function properly. The `include` statement must appear at the top of every template file that uses an Ektron CMS400.NET custom ASP function.

**IMPORTANT!**

`ContentWS.asmx` is the web services (WSDL) file that the custom ASP functions use to talk to **`Ektron CMS400.NET`**. The WSDLpath located in the `applicationAPI.asp` file must be set to location of the WSDL file. The default WSDLpath is `http://localhost/CMS400Example/WorkArea/WebServices/ContentWS.asmx`. If you have changed the location of `ContentWS.asmx` you need to update the WSDLpath. For example, if your site is `www.myektronsite.com`, the WSDLpath would be `http://www.myektronsite.com/WorkArea/WebServices/ContentWS.asmx`.

To customize Ektron CMS400.NET, you need to be familiar with these tags.

| Function | Description | More Information |
|---|---|---|
| **Calendar Functions** | | |
| Calendar | Displays a calendar with different display types. For example, a daily view calendar. | "Calendar Function" on page 353 |
| Event Calendar | Displays a month view event calendar | "Event Calendar Function" on page 350 |
| **Collection Functions** | | |
| Collection | Displays links generated by collections | "Collection Function" on page 328 |

| Function | Description | More Information |
|---|---|---|
| CollectionXML | Creates XML for a collection by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed. | "XML Collection Function" on page 343 |
| Random Content | Randomly displays the content of a collection item | "Random Content" on page 348 |
| Random Summary | Randomly displays the summary of a collection item | "Random Summary" on page 349 |
| RSS Collection | Creates an RSS feed out of the items in a collection. | "RSS Collection Function" on page 341 |
| **Content Functions** | | |
| Content Block | Displays and manages a content block | "Content Block Function" on page 278 |
| Form | Displays a content block associated with a form | "Form Function" on page 357 |
| Multipurpose | Displays either a content block or a form | "MultiPurpose Function" on page 360 |
| **List Summary Functions** | | |
| Archive ListSummary | Displays summaries of archived content blocks | "Archive List Summary Function" on page 317 |
| List Summary | Displays a list of content block summaries | "List Summary Function" on page 304 |
| List Summary XML | Creates XML for a list summary by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed. | "XML List Summary Function" on page 312 |
| RSS List Summary | Creates an RSS feed out of the items in a list summary. | "RSS Summary Function" on page 322 |
| Single Summary | Displays the summary for a content block | "Single Summary Function" on page 318 |

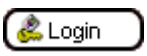
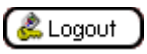
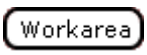
| Function | Description | More Information |
|---|---|---|
| **Login Functions** | | |
| Login | Adds the login or logout button | ”Login Function” on page 276 |
| **Menu Functions** | | |
| DropHTMLMenu | Used to display a DHTML menu | ”DHTML Menus” on page 370 |
| GenerateMenu | Works with DropHTML menu to display DHTML menus | ”DHTML Menus” on page 370 |
| XML Menu | Creates XML for a menu by wrapping tags around all of the menu items properties. | ”XML Menu” on page 375 |
| **Metadata Functions** | | |
| Metadata | Displays the metadata for the current Web page | ”Metadata Function” on page 361 |
| **Search Functions** | | |
| Search | Inserts a text box with a search button | ”Search Functions” on page 288 |
| Search Display | Creates an area to display search results | ”Search Display Function” on page 299 |
| **Site Language Functions** | | |
| ShowSiteLanguage | Lets a site visitor select a language in which to view site. | ”Show Site Language Function” on page 369 |

# Login Function

The login function adds a login button on the template when displayed in a browser. Here it the format of the login function.

```
<% ecmLogin %>
```

This tag is responsible for displaying the following buttons on the Web page.

| Button | Name | Description |
|---|---|---|
|  | Login | When user is not logged into Ektron CMS400.NET, this button appears. Click this button to let the user log in to the application. |
|  | Logout | Once logged in, this button replaces the login button to allow the user to log out from the Ektron CMS400.NET Web site. |
|  | Workarea | When logged in, this button appears under the logout button allowing the user to access their workarea. |
| | Preview On | Lets the user to preview the entire Web site as if all checked-in content was published. |
| | Preview Off | Turns off preview mode. |

# Login Tag Example

The following example places a login button in a table cell inside an asp template.

```
<tr>
    <td>
        <% ecmLogin %>
    </td>
</tr>
```

When inserted properly, a login button appears on the Web page.

Clicking the login button opens a login window in which a user enters the username and password. Upon successful completion, the user is logged in to the Ektron CMS400.NET Web site.

# Placing Your Login Button

You can add as many login buttons to a template as you like. You might want to include a login button on each template, or have a special Web page called login.asp, from which users can log in without the public being able to access the login page.

# Content Block Function

The content block tag displays a content block on an Ektron CMS400.NET Web page. There are two functions to choose from:

- ecmContentBlock
- ecmContentBlockEX

ecmContentBlock allows you display a content block without having to define an XSLT.

With ecmContentBlockEX, you can display XML content by defining an XSLT or applying one of the predefined XSLTs located in **Ektron CMS400.NET**. For more information on predefined XSLTs, see the **Administrator's Manual** > "Managing XML" > "Adding an XML Configuration" > "Adding an XML Configuration Using External XML Files".

Here is the format for the ecmContentBlock function.

```
<%ecmContentBlockEX     ID          %>
```

Here is the format for the ecmContentBlockEX function.

```
<%ecmContentBlockEX     ID,
                        "XSLT",
                        Override    %>
```

**NOTE** As an alternative, you can use the multipurpose function, which can display either a content block or a content block associated with a form. For more information, see "MultiPurpose Function" on page 360.

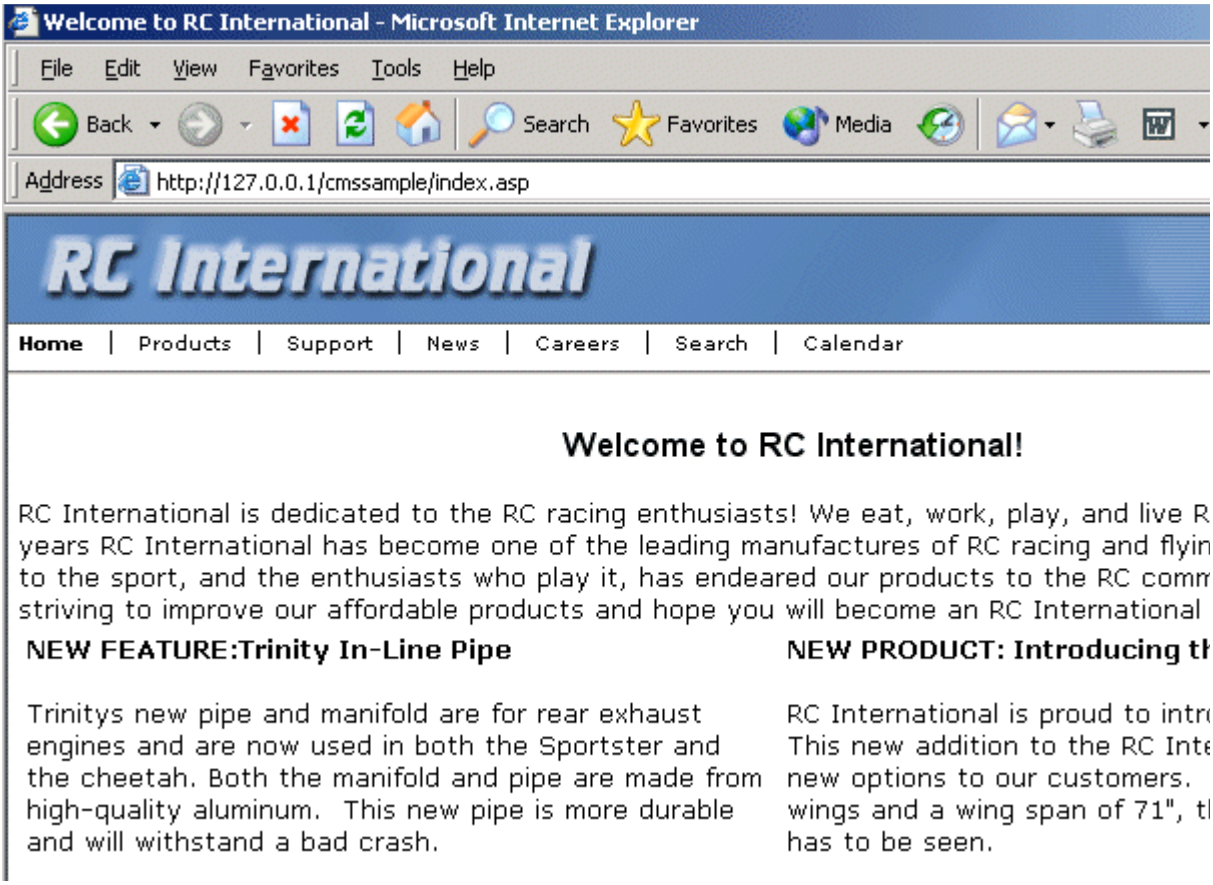
The table below explains the attributes of the ecmContentBlock function.

| Attribute | Description | Required |
|---|---|---|
| ID | ID number assigned of the content block that will appear | Yes |

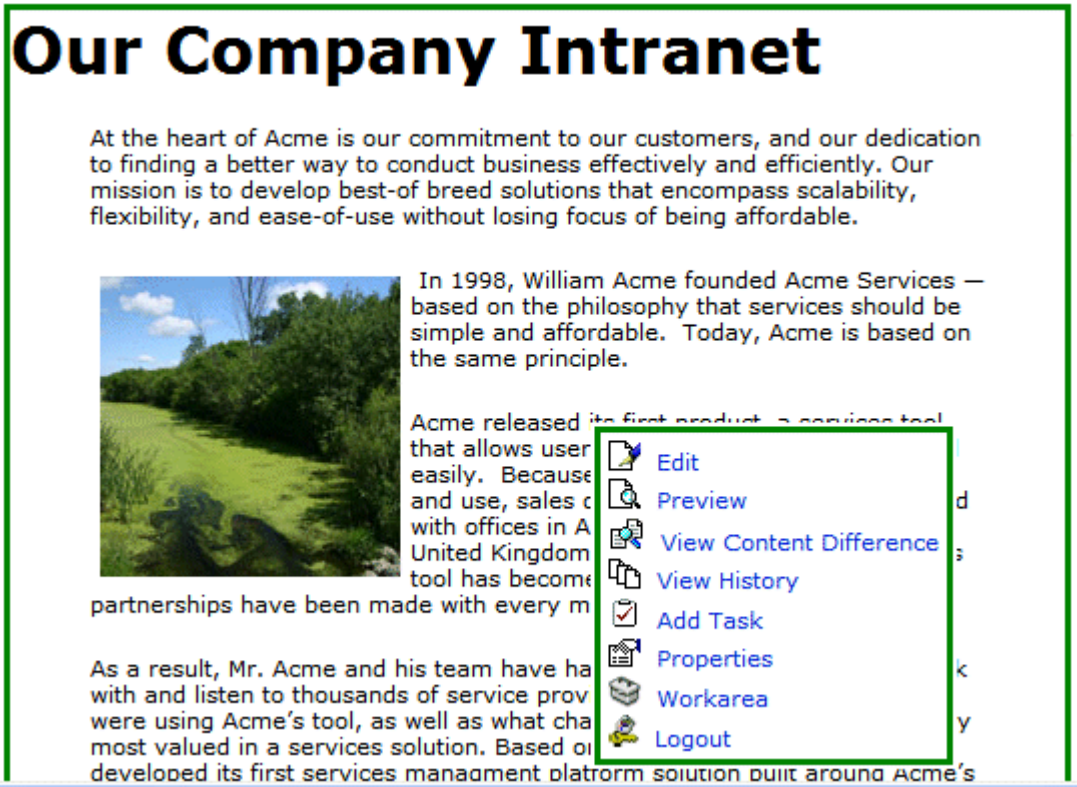
The table below explains the attributes of the ecmContentBlockEX function.

| Attribute | Description | Required |
|---|---|---|
| ID | ID number assigned of the content block that will appear | Yes |
| XSLT | File name and path to a valid XSLT to transform the XML content. *See Also:* "The Function's Arguments" on page 285 | No |
| Override | If no XSLT is specified, then enter 1 to use XSLT1, 2 for XSLT2, or 3 for XSLT3. *See Also:* "The Function's Arguments" on page 285 | No |

In normal view, these tags retrieve the content block from the database and displays it in the template.


Welcome to RC International - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://127.0.0.1/cmssample/index.asp
RC International
Home | Products | Support | News | Careers | Search | Calendar
Welcome to RC International!
RC International is dedicated to the RC racing enthusiasts! We eat, work, play, and live R
years RC International has become one of the leading manufactures of RC racing and flyin
to the sport, and the enthusiasts who play it, has endeared our products to the RC comm
striving to improve our affordable products and hope you will become an RC International
NEW FEATURE:Trinity In-Line Pipe
Trinitys new pipe and manifold are for rear exhaust engines and are now used in both the Sportster and the cheetah. Both the manifold and pipe are made from high-quality aluminum. This new pipe is more durable and will withstand a bad crash.
NEW PRODUCT: Introducing th
RC International is proud to intr
This new addition to the RC Inte
new options to our customers.
wings and a wing span of 71", t
has to be seen.

If a user is logged into Ektron CMS400.NET, the content block tag displays a border around the content block and when they right click on the content, a menu is provided for managing it.


Our Company Intranet
At the heart of Acme is our commitment to our customers, and our dedication to finding a better way to conduct business effectively and efficiently. Our mission is to develop best-of breed solutions that encompass scalability, flexibility, and ease-of-use without losing focus of being affordable.
In 1998, William Acme founded Acme Services — based on the philosophy that services should be simple and affordable. Today, Acme is based on the same principle.
Edit
Preview
View Content Difference
View History
Add Task
Properties
Workarea
Logout

# Content Block Examples

**Ektron CMS400.NET** has two types of content blocks.

| Type | Displays | More information |
|---|---|---|
| Static | One content block on the Web page | "Static Content Block" on page 282 |
| Dynamic | A content block whose ID is passed through the URL | "Dynamic Content Block" on page 283 |

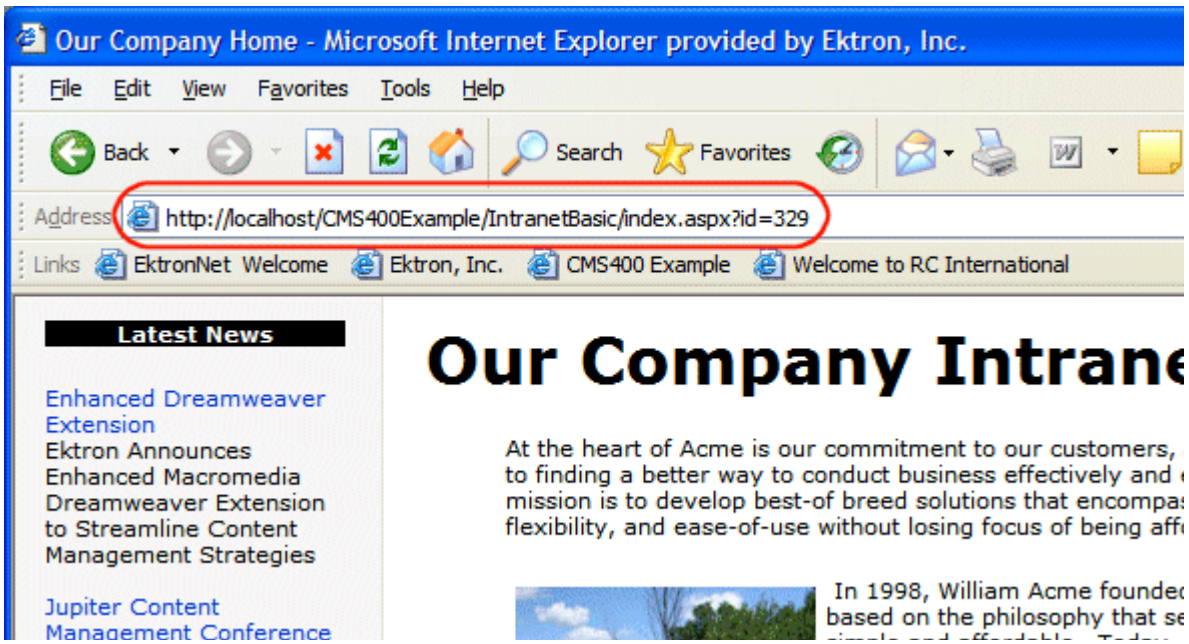
In addition, a content block can contain XHTML or XML content. For more information, see "XML Content Block" on page 284.

## Static Content Block

The following illustrates how to insert a static content block on a template.

```
<tr>
    <td>
        <% ecmContentBlock(329) %>
    </td>
</tr>
```

In the example above, the function retrieves the content block with an id of 329 from the database to the browser.

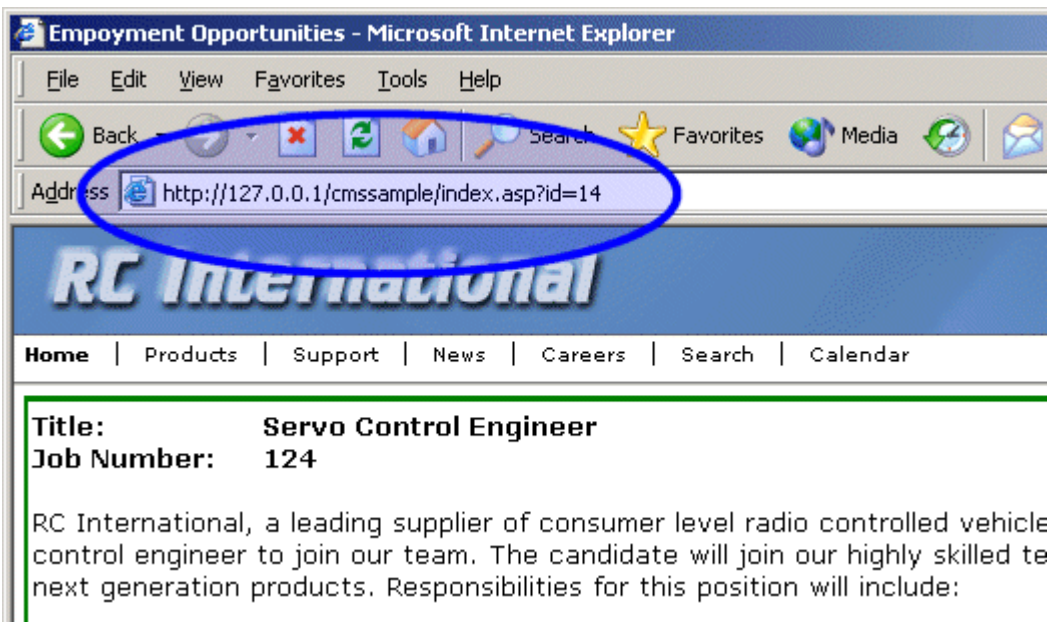

Our Company Home - Microsoft Internet Explorer provided by Ektron, Inc.
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/CMS400Example/IntranetBasic/index.aspx?id=329
Links EktronNet Welcome Ektron, Inc. CMS400 Example Welcome to RC International
Latest News
Enhanced Dreamweaver Extension
Ektron Announces Enhanced Macromedia Dreamweaver Extension to Streamline Content Management Strategies
Jupiter Content Management Conference
Our Company Intrane
At the heart of Acme is our commitment to our customers,
to finding a better way to conduct business effectively and
mission is to develop best-of breed solutions that encompas
flexibility, and ease-of-use without losing focus of being aff
In 1998, William Acme founde
based on the philosophy that se

## Dynamic Content Block

The following example shows the content block tag where the id is being passed as a URL parameter. Use this format with dynamic template.

```
<tr>
    <td>
        <% ecmContentBlock(request.QueryString("id")) %>
    </td>
</tr>
```

When a user accesses this page, it adds the content block ID number to the tag's template, and the content block with the corresponding ID appears on the Web page in the specified template.


Empoyment Opportunities - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Favorites
Media
http://127.0.0.1/cmssample/index.asp?id=14
RC International
Home | Products | Support | News | Careers | Search | Calendar
Title: Servo Control Engineer
Job Number: 124
RC International, a leading supplier of consumer level radio controlled vehicle
control engineer to join our team. The candidate will join our highly skilled te
next generation products. Responsibilities for this position will include:

The `id=` convention is used by **Ektron CMS400.NET** to generate quicklinks when content blocks are created. (For more information, see the **Ektron CMS400.NET** Administrator manual section

"Managing Content" > "Forming URLs For Ektron CMS400.NET Documents.")

The dynamic `ecmContentBlock` function is almost the same as the static one (see "Static Content Block" on page 282). The only difference is the parameter, which makes the function dynamic. The parameter uses a query string call (`Request.QueryString`) to read the URL from the browser's address bar. It treats everything after the question mark as a list of key/value pairs separated by ampersands. So, `Request.QueryString` passes the key as a parameter and returns its associated value. For example, in URL `<http://localhost/CMS400Example/index.asp?id=1&LangType=1036>`, the following QueryString calls return these values.

| QueryString Call | Return value |
|---|---|
| Request.QueryString("id") | 1 |
| Request.QueryString("LangType") | 1036 |

## XML Content Block

Here is an example of displaying an XML content block. Note that you specify a content block and an XSLT, which determines how the content block is displayed.

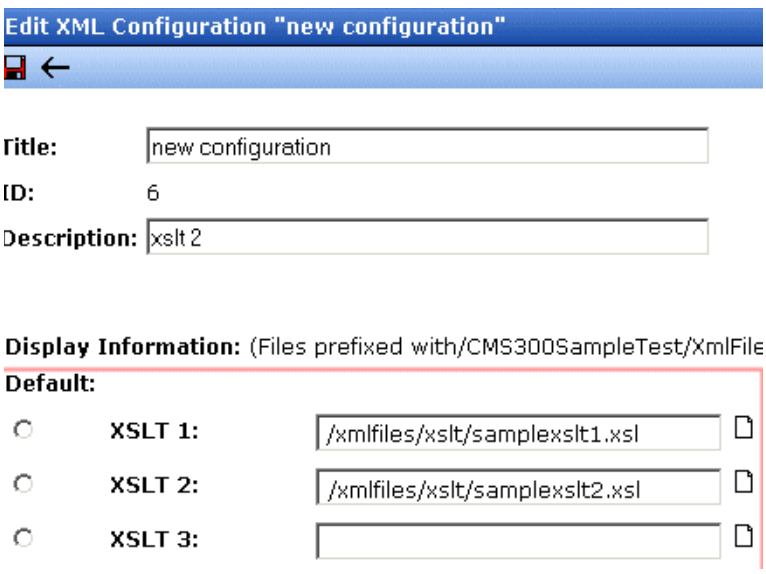
```
<tr>
    <td>
        <% ecmContentBlockEx 13, "", 1 %>
    </td>
</tr>
```

This example displays the content block ID=13, using XSLT1 as the display XSLT.

### *The Function's Arguments*

The ecmContentBlockEx takes three arguments. The first argument identifies the content block to display. The second one is optional and, if used, specifies an external XSLT file.

If the second argument does not exist, the third argument specifies an XSLT identified in the Edit XML Configuration screen. The following table provides additional detail about the second and third arguments.

| To use this display XSLT | Enter this for argument 2 | Enter this for argument 3 |
|---|---|---|
| The default XSLT specified in the Edit XML configuration screen (illustrated below). Note that, in this example, the **XSLT Packaged** option is the default XSLT since it is selected.<br>  | "" | none |

| To use this display XSLT | Enter this for argument 2 | Enter this for argument 3 |
|---|---|---|
| **XSLT Packaged:** from the Edit XML configuration screen (that is, the XSLT created in the Data Designer)<br><br>Note: If an XSLT package has not been created, and zero (0) is the third argument, the default XSLT specified in the Edit XML configuration screen is used. | “” | 0 |
| **XSLT 1** from the Edit XML configuration screen | “” | 1 |
| **XSLT 2** from the Edit XML configuration screen | “” | 2 |
| **XSLT 3** from the Edit XML configuration screen | “” | 3 |
| An absolute or relative path to an XSLT | An external XSLT file not specified in the Edit XML Configuration screen.<br>For example<br>`<% ecmContentBlockEx 13, "samplexslt.xsl"%>` | does not matter - if a value exists in argument 2, argument 3 is ignored |

*See Also:* The “Managing XML” chapter of the **Ektron CMS400.NET** Administrator manual

## Dynamic and Static Combination

Here is an example of how you can use the two different content tags in the same table cell by using an IF statement

```
<tr>

    <td>

        <% if request.QueryString("id")<> "" then

            ecmContentBlock(request.QueryString("id"))else

            ecmContentBlock(14) end if %>

    </td>

<tr>
```

In this sample code, when a person accesses this page without passing through the content's ID number, the content with an ID=14 will be displayed. If you pass a content id through the URL like **index.asp?id=4,** the content block ID=4 will be displayed.

# Search Functions

The search tag is used to display a text box, with a search button on the template. When seen on the template of a Web page, users can enter text and click on the search button to search through all the HTML content on the Web site. To create a search, Ektron has created two search functions:

- ecmSearch
- ecmSearchWithFormTag

Both functions have the same parameters and perform the same search. The only difference is ecmSearchWithFormTag automatically inserts form tags, ecmSearch does not.

The following is an example of form tags.

```
<form name="ecmsearch" method="post" action='search-
display.asp'>

</form>
```

When using ecmSearch, form tags need to be added manually.

Shown here is the format for the ecmSearch and ecmSearchWithFormTag:

```
<% ecmSearch
or
ecmSearchWithFormTag    "Starting Folder",
                        Recursive,
                        "Target Page",
                        Text Box Size,
                        Maximum Characters,
                        "Button Image Source",
                        "Button Text",
                        "Font Face",
                        "Font Color",
                        "Font Size",
                        Horizontal,
                        "Spare"
                                                %>
```

The following table provides information about all the attributes for the search tag.

| Attribute | Description | Required ? | For more information, see |
|---|---|---|---|
| Starting Folder | The folder you wish to search. | Yes | ”Starting Folder” on page 292 |
| Recursive | Whether to search subfolders. 0=No, 1 = Yes | Yes | ”Recursive” on page 292 |
| Target Page | The Web page to output the search results. | Yes | ”Target Page” on page 293 |
| Text Box | The size of the text box for user input. | Yes | ”Text Box” on page 293 |

| Attribute | Description | Required ? | For more information, see |
|---|---|---|---|
| Maximum Characters | The maximum number of characters that the input text box can accept. | No | ”Maximum Characters” on page 294 |
| Button Image Source | The image location for the search button. Blank (“”) = standard submit button. | No | ”Button Image Source” on page 294 |
| Button Text | The text used for the button if the standard submit button is chosen, or alternative text for the button if an image is chosen. | No | ”Button Text” on page 295 |
| Font Face | The font of the search tag text. | No | ”Font Face” on page 296 |
| Font Color | The font color of the search tag text. Blank (“”) = Black. | No | |
| Font Size | The font size of the search tag text. | No | ”Font Size” on page 297 |
| Horizontal | Determines if the search options are in a row or a column. 1 = horizontal, 0 = vertical | Yes | ”Horizontal” on page 298 |
| Spare1 | Currently not used. | Must be “” | ”Spare 1” on page 298 |

# Search Tag Example

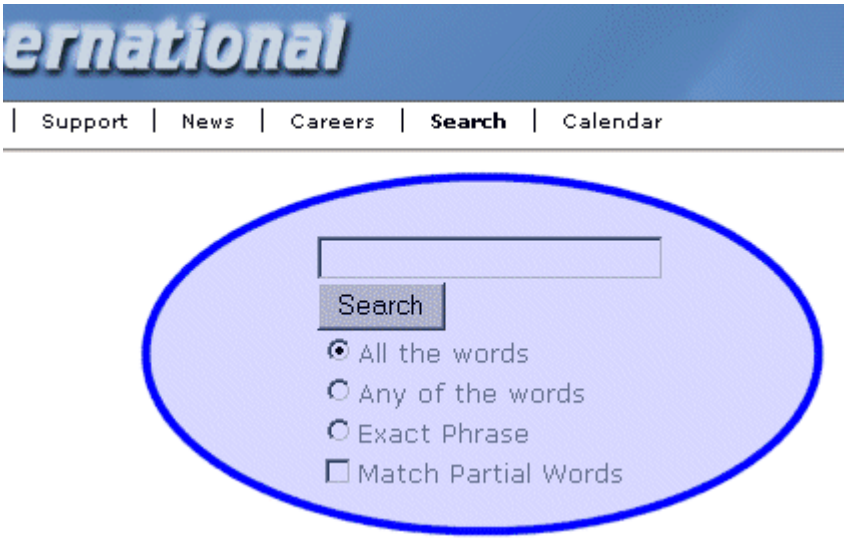
The following example places a text field and a search button in a table cell inside an ASP template.

```
<tr>
    <td>
        <% ecmSearchWithFormTag "\",1,"search-
        display.asp",25,200,"","Search","Verdana",
        "#808080","2",0,"" %>
    </td>
</tr>
```

The previous example has the following properties:

- Searches the root folder, in this case the `\content` folder.
- Search is recursive.
- The target page is “search-display.asp”.
- The size of the text box is 25.
- The search text box allows up to 200 characters.
- No image is used for a button. Instead, a standard submit button is used.
- The text on the submit button is “Search”.
- The search options appear in Verdana with a color of #808080 and a size of 2pt.
- The search options are arranged vertically.

When added to a template, and visited, the search tag will appear similar to the following:




# Search Tag Attributes

The search tag's attributes are described below.

## Starting Folder

In this attribute, specify the content folder where the search begins. There are two options for this attribute.

| Value | Search begins in |
|---|---|
| "\" | root folder |
| "\folder path" | specified folder |

## Recursive

When you setup the search to be recursive, you are allowing the search to search all the sub-folders associated with the starting folder you had specified. You have two options for this attribute:

| Value | Search begins in |
|---|---|
| 1 | Recursive search (will search sub-folders of the starting folder). |
| 0 | Non-Recursive search (will only search starting folder). |

## Target Page

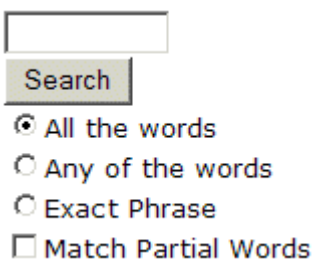
The page that displays the search results. You may enter any form of a URL for this page.

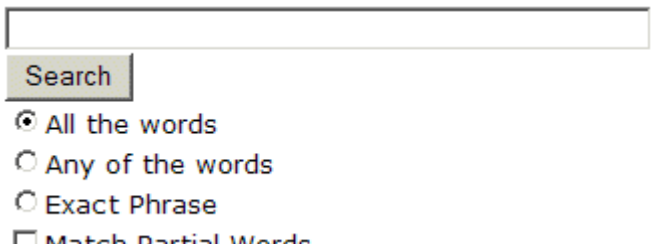
**NOTE** The target page *must* include the searchdisplay function to display search results.

To use the same page, enter a pound sign (#) into the field.

## Text Box

This attribute sets the size of the search text box. It can be any number, depending on the size of the search text box that you want.

| Text Box Size | Web Page View |
|---|---|
| 10 |  |

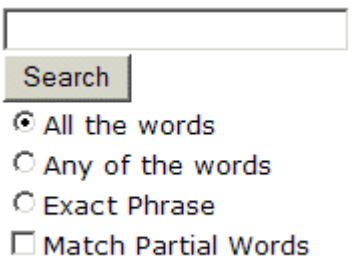
| Text Box Size | Web Page View |
|---|---|
| 50 |  |

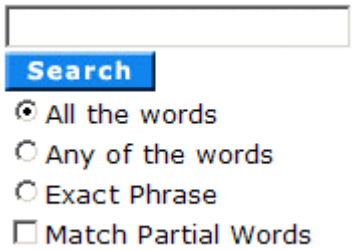
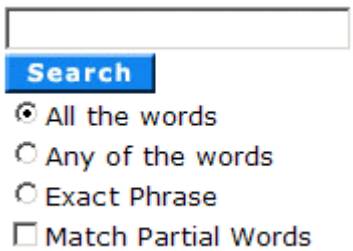
## Maximum Characters

This attribute determines the maximum number of characters that a user can enter for search text.

## Button Image Source

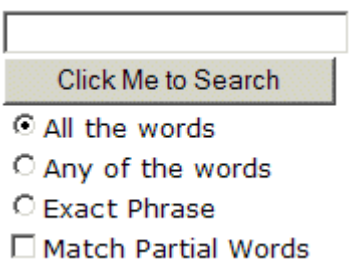
To use an image for the search button, enter the image's filename and path.

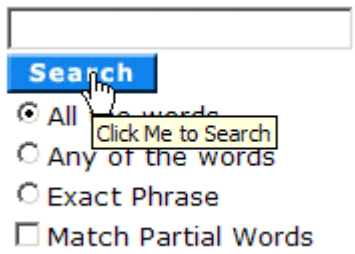
| Button Image Source | Web Page View |
|---|---|
| "" |  |

| Button Image Source | Web Page View |
|---|---|
| "search.jpg" |  |
| "http://www.ektron.com/images/search.jpg" |  |

## Button Text

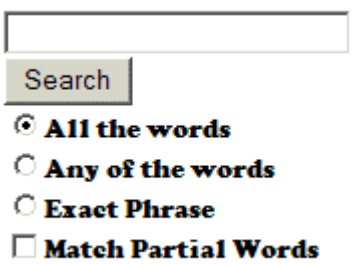
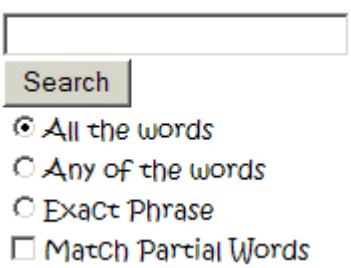
If no search button image is specified, the text in this field appears on the button. If an image is being used for the search button, this text is the image's alt text.

| Button Text | Web Page View |
|---|---|
| "Click Me to Search" |  |

| Button Text | Web Page View |
|---|---|
| "Click Me to Search" |  |

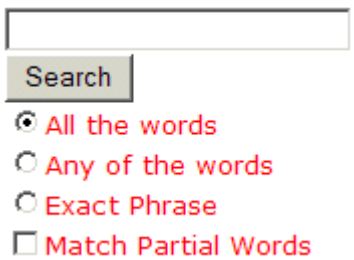
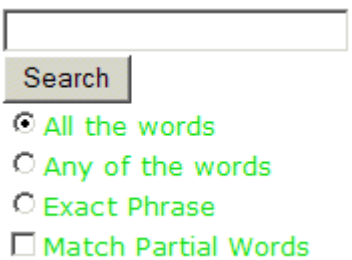
## Font Face

This font is used for the search options displayed on the Web page.

| Font Face | Web Page View |
|---|---|
| Cooper Black |  |
| Kristen ITC |  |

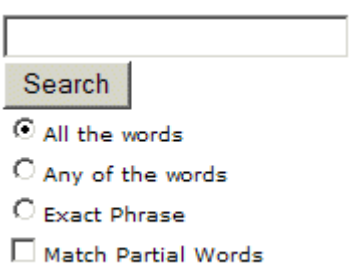
## Font Color

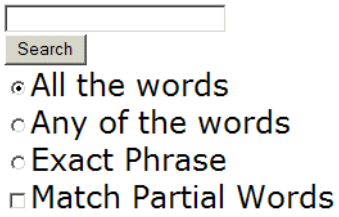
The color of the text in the search tag. You may enter the color name or a hexidecimal value of the color.

| Font Color | Web Page View |
|---|---|
| Red |  |
| #14DA14 |  |

## Font Size

This is the size of the text on the search tag.

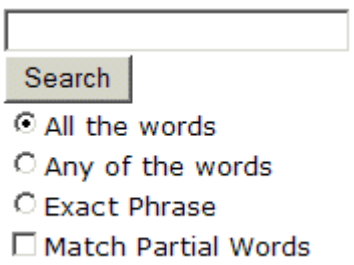
| Font Size | Web Page View |
|---|---|
| 1 |  |

| Font Size | Web Page View |
|---|---|
| 5 |  |



## Horizontal

This attribute determines whether the search options are displayed vertically or horizontally.

| Attribute | Web Page View |
|---|---|
| 0<br>(vertical) |  |
| 1<br>(horizontal) | |



## Spare 1

At the end of the Search tag is a attribute called Spare 1. This attribute has not yet been assigned an function. As of now, when you enter a search tag, you must pass this attribute “”.

# Search Display Function

The search display custom tag creates the area where search results appear after a search is executed.

Here is the format of the ecmSearchDisplay tag.

```
<% ecmSearchDisplay     "MaxNumberReturn",
                        "StyleInfo",
                        ShowDate                %>
```

Below is a description of the tag's attributes.

| Attribute | Description | Required | For more information, see |
|---|---|---|---|
| Max Number of content blocks Returned | Maximum number of content blocks returned. (0=Unlimited) | Yes | "Max Number of Content Blocks Returned" on page 302 |
| Style Info | An HTML style string used within the HTML "hyperlink" generated by the search output. Can control font, color, size, etc. | No | "Style Info" on page 303 |
| ShowDate | Displays the last modified date of the content block. 0=No 1=Yes | No | "Show Date" on page 303 |

# Search Display Example

Insert the following on the page that displays search results. You may place it on the same page as the search tag or a separate page.

```
<tr>
    <td>
        <% ecmSearchDisplay 0,"","1" %>
    </td>
</tr>
```

This search display example has the following properties:

- Returns unlimited results when the search is performed.
- Contains no style information.
- Shows the last modified date of the content.

When the tag is added to a template, and a search is performed, the results appear similar to the following.

## Search Display with Default Content

You can also set up the search display tag to display a default content block if a search is not performed. Then, if a search is

performed, the results replace the content block. The sample code below does that:

```
<tr>
    <% if (Request.Form("ecmsearchtype") <> "") then %>
        <td>
            <% ecmSearchDisplay 0,"" %>
        </td>
    <% else %>
        <td>
            <% ecmContentBlock(1) %>
        </td>
    <% end if %>
</tr>
```

## Search Display Attributes

Listed below are the attributes for the SearchDisplay tag.

## Max Number of Content Blocks Returned

This attribute determines the number of search results displayed on a page. You may enter any integer. To display unlimited results, enter 0 (zero).

| Max Returns | Web Page View |
|---|---|
| 0 | **Search Results**<br><br>RC Sportster<br>RC Redstar<br>RC Lilly<br>RC Cheetah<br>Products Page<br>RC International Ships RC Sportster<br>RC International Hosts RC Conference<br>RC International Announces the RC Cheetah |
| 3 | **Search Results**<br><br>RC Sportster<br>RC Redstar<br>RC Lilly |

## Style Info

An HTML style string used within the HTML “hyperlink” generated by the search output. Can control font, color, size, etc.

| Style Info | Web Page View |
|---|---|
| ““ | **Search Results**<br>RC Sportster<br>RC Redstar<br>RC Lilly<br>RC Cheetah<br>Products Page<br>RC International Ships RC Sportster<br>RC International Hosts RC Conference<br>RC International Announces the RC Cheetah |
| "font-family:arial; font-weight:bold; background-color:#cccccc; border:solid blue 1pt; margin:2px; width:50%;" | **Search Results**<br>RC Sportster<br>RC Redstar<br>RC Lilly<br>RC Cheetah<br>Products Page<br>RC International Ships RC Sportster<br>RC International Hosts RC Conference<br>RC International Announces the RC Cheetah |

## Show Date

This attribute determines if the content block’s last modified date appears next to the title. This is useful when determining which content block is the most recent. You have two choices.

| Attribute | Web Page View |
|---|---|
| 0 | **Search Results**<br><br>RC Sportster<br>RC Redstar<br>RC Lilly<br>RC Cheetah<br>Products Page<br>RC International Ships RC Sportster<br>RC International Hosts RC Conference<br>RC International Announces the RC Cheetah |
| 1 | **Search Results**<br><br>RC Sportster *(8/27/2002 7:52:40 PM)*<br>RC Redstar *(8/27/2002 7:52:01 PM)*<br>RC Lilly *(8/27/2002 7:51:26 PM)*<br>RC Cheetah *(8/27/2002 7:50:53 PM)*<br>Products Page *(8/27/2002 7:50:09 PM)*<br>RC International Ships RC Sportster *(8/27/2002 2:46:12 PM)*<br>RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)*<br>RC International Announces the RC Cheetah *(8/27/2002 2:43:01 PM)* |

# List Summary Function

The list summary function is used to display the list of content summaries on a Web page.

Shown here is the format of the list summary function:

```
<% ecmListSummary    "Folder",
                     Recursive,
                     ShowSummary,
                     "StyleInfo",
                     "OrderBy, Asc/Desc",
                     "ShowInfo",
                     MaxNumber,
                     "ObjType",
                     "OptionList"                %>
```

A list and description of the attributes are shown in this table:

| Attribute | Description | Options |
|---|---|---|
| Folder | This is the folder the summary information will come from. | Any folder |
| Recursive | Allow the tag to get summary info from the child folders. | 0 = Not Recursive<br>1 = Recursive |
| ShowSummary | Do you want the content summary to be displayed? | 0 = No<br>1 = Yes |
| StyleInfo | Inline style information for the generated text hyperlink. | Any string |
| OrderBy, ASC/DESC | Order the hyperlinks by: | “Title”<br>“DateModified”<br>“DateCreated”<br>“Startdate”<br>“AuthorFname”<br>“AuthorLname” |
| | Order the hyperlinks in ascending or descending order. | “Title,asc”<br>“Title,desc” |
| ShowInfo | Information that will appear to the right of the hyperlink. | “DateModified”<br>“DateCreated”<br>“AuthorFname”<br>“AuthorLname” |

| Attribute | Description | Options |
|---|---|---|
| MaxNumber | Maximum number of summaries returned. | 0 = Unlimited |
| ObjType | Determines whether the list summary is for the folder or a single piece of content.<br>For example:<br>If the folder property = \books and “Folder” is chosen as the ObjType, the list summary for the folder with books appears.<br>If the folder property = 1 and you choose “Content” as the ObjType, a single summary for content with an id = 1 appears.<br>*See Also:* ”Single Summary Function” on page 318<br><br>The default ObjType is “Folder” when left blank. | ““<br>“Folder“<br>“Content” |
| OptionList | Not yet implemented | MUST BE ““ |

# List Summary Example

Shown below is an example of the ListSummary function in an ASP Web page.

```
<TR>
    <TD>
        <% ecmListSummary "\Marketing\News",0,1,"",
            "Title,asc","DateModified",0,"","" %>
    </TD>
</TR>
```

The previous example has the following properties:

- Displays summaries from the \Marketing\News folder.
- Not recursive.
- The summaries will be shown.
- No style was applied to the hyperlinks.

- The results will be displayed in ascending order by title.
- The date modified of each content block will be displayed.
- Unlimited results will be shown.

Here is how the summaries will appear on the web page.

Latest News
RC International Announces the RC Cheetah (8/27/2002 2:43:01 PM)
RC International. a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah.
RC International Hosts RC Conference (8/27/2002 2:44:44 PM)
RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference in Atlanta, Georgia.

# List Summary Attributes

Listed below are all the different attributes and descriptions about them.

## Starting Folder

In this attribute, you will need to specify which content folder you will want to get all the summaries from. There are two options for this attribute:

| Value | Description |
|---|---|
| “” | By leaving the value blank, all the summaries from the root content folder will be displayed. |
| “\Folder Path | You can specify a specific folder to display the summaries from by inserting the folder path |

## Recursive

When you setup the search to be recursive, you are allowing the search to search all the subfolders associated with the starting folder you had specified. You have two options for this attribute:

| Value | Description |
|---|---|
| 1 | Recursive - summaries in the starting folder's subfolders will also be displayed. |
| 0 | Non-recursive - only summaries in the starting folder will be displayed. |

## Show Summary

The show summary attribute displays or suppresses the summary in the summary list. You have two options for this attribute.

| Attribute | Web Page View |
|---|---|
| 0 | RC International Ships RC Sportster<br><br>RC International Hosts RC Conference<br><br>RC International Announces the RC Cheet |
| 1 | RC International Announces the RC Cheetah *(8/27/2002 2:43:01 PM)*<br>RC International, a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah.<br><br>RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)*<br>RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference in Atlanta, Georgia. |

## Style Info

In this field, you may enter inline style information for the generated text hyperlinks displayed on the screen.

| Style Info | Web Page View |
|---|---|
| "" | RC International Announces the RC Cheetah *(8/27/2002 2:43:01* RC International, a leading RC racing company, has announced th availability of its new flagship product, the RC Cheetah.<br><br>RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)* RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference Atlanta, Georgia. |
| "font-family:arial; font-weight:bold; background-color:#cccccc; border:solid blue 1pt; margin:2px; width:100%;" | **RC International Announces the RC Cheetah**<br>RC International, a leading RC racing company, has announced tl availability of its new flagship product, the RC Cheetah.<br><br>**RC International Hosts RC Conference**<br>RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference Atlanta, Georgia. |

## Order By

This attribute will sort the summary results by different options. There are five different ways to sort the summaries:

| Value | Summaries sorted by |
|---|---|
| Title | title |
| DateModified | date modified |
| Date Created | date created |
| Startdate | The Start Date assigned to the content block when created. |

| Value | Summaries sorted by |
|---|---|
| AuthorFname | author's first names |
| AuthorLname | author's last name |

### *Ascending/Descending*

This attribute is an addition to the previous OrderBy attribute. By default, the list of the summaries are descending, meaning they will follow the path of A-B-C, or 1-2-3. When set to ascending, they will order by Z-Y-X or 3-2-1.You will have the option to re arrange these to display either in ascending or descending order.

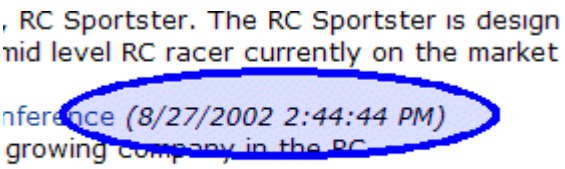
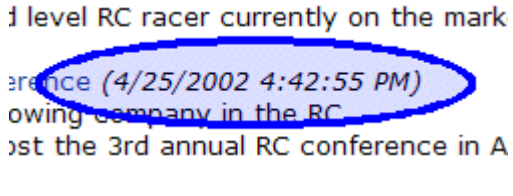
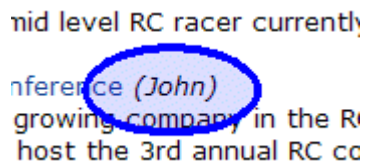
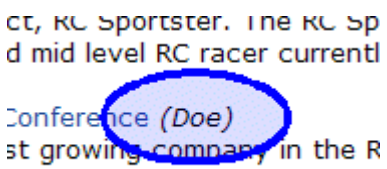
To do this, add the following to the OrderBy attribute:

| Value | Description |
|---|---|
| asc, 1 | This will order the Summaries in an ascending order |
| desc, 0 | This will order the Summaries in a descending order. This is the default setting. |

By using the ascending/descending option, you will have the option to set the order of content blocks from most recent to oldest, or vice versa. As well as ordering the list by the user's first names or last names in ascending or descending order.

## Show Info

The show info field determines what sort of information is displayed to the right of the hyperlinked title. There are four different options that you may choose from:

| Value | Description | Web Page View |
|---|---|---|
| DateModified | Displays the date the content was last modified. |  , RC Sportster. The RC Sportster is design<br>mid level RC racer currently on the market<br>nference (8/27/2002 2:44:44 PM)<br>growing company in the RC<br>host the 3rd annual RC conference in Atla |
| ***DateCreated | Displays the date when the content was first created. |  d level RC racer currently on the mark<br>erence (4/25/2002 4:42:55 PM)<br>owing company in the RC<br>ost the 3rd annual RC conference in A |
| AuthorFname | Displays the first name of the user who last edited the content block. |   |
| AuthorLname | Displays the last name of the user who last edited the content block. |  ct, RC Sportster. The RC Sp<br>d mid level RC racer currentl<br>Conference (Doe)<br>st growing company in the R<br>to host the 3rd annual RC c |

## Max Number

This attribute determines how many summaries appear on a page. Enter the number of summaries to be displayed. To display all summaries, enter 0 (zero) and you will display unlimited results.

# XML List Summary Function

The XML List Summary function wraps XML around the properties of a list of content thats created as a list summary in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the list summary information. The ecmListSummaryXML function allows you to easily define how the list summary data appears on the Web page.

Shown here is the format of the ecmListSummaryXML function.

```
<% ecmListSummaryXML   “Folder”,
                       “Recursive”,
                       “OrderBy”,
                       “MaxNumber”,
                       “ObjType”,
                       “SummaryType”,      %>
```

The ecmListSummaryXml function attributes are explained below.

| Attribute | Description | Options |
|---|---|---|
| Folder | This is the folder the summary information will come from.<br>*See Also:* ”Starting Folder” on page 307 | Any folder |
| Recursive | Allow the tag to get summary info from the child folders.<br>*See Also:* ”Recursive” on page 308 | 0 = Not Recursive<br>1 = Recursive |
| OrderBy, ASC/ DESC | Order the hyperlinks by:<br>*See Also:* ”Order By” on page 309 | “Title”<br>“DateModified”<br>“DateCreated”<br>“Startdate”<br>“AuthorFname”<br>“AuthorLname” |
| | Order the hyperlinks in ascending or descending order. | “Title,asc”<br>“Title,desc” |

| Attribute | Description | Options |
|---|---|---|
| MaxNumber | Maximum number of summaries returned.<br>*See Also:* "Max Number" on page 311 | **0** (Zero) = Unlimited |
| ObjType | Determines whether the list summary is for the folder or a single piece of content.<br>For example:<br>If the folder property = \books and "Folder" is chosen as the ObjType, the list summary for the folder with books appears.<br>If the folder property = 1 and you choose "Content" as the ObjType, a single summary for content with an id = 1 appears.<br>*See Also:* "Single Summary Function" on page 318<br><br>The default ObjType is "Folder" when left blank. | ""<br>"Folder"<br>"Content" |

When using the XML list summary function, the following tags are created around the properties of each item in the list summary.

| XML Tags | Description |
|---|---|
| <Content></Content> | Declares a new section of content in the XML collection. |
| <ID></ID> | The content block ID.<br>For example:<br>`<ID>27</ID>` |
| <Title></Title> | The title of the content block.<br>For example:<br>`<Title>HTML for the World Wide Web with XHTML and CSS</Title>` |
| <QuickLink></QuickLink> | The quicklink associated with the content block.<br>For example:<br>`<QuickLink>/CMS400Example/collection.aspx?id=27</QuickLink>` |
| <Teaser></Teaser> | The summary of the content block.<br>For example:<br>`<Teaser>The easiest HTML book weve seen that still manages to be comprehensive.</Teaser>` |

| XML Tags | Description |
|---|---|
| <StartDate></StartDate> | The content block's start date formatted as a .NET date type<br>For example:<br>`<StartDate>1/1/0001 12:00:00 AM</StartDate>` |
| <DateModified></DateModified> | The last date the content block was modified.<br>For example:<br>`<DateModified>1/19/2005 4:18:06 PM</DateModified>` |
| <EndDate></EndDate> | The date the content block stops running on the Web site<br>For example:<br>`<EndDate>12/31/9999 12:00:00 AM</EndDate>` |
| <LastEditorFname></LastEditorFname> | The first name of the last person who edited the content block.<br>For example:<br>`<LastEditorFname>John</LastEditorFname>` |
| <LastEditorLname></LastEditorLname> | The last name of the last person who edited the content block.<br>For example:<br>`<LastEditorLname>Johnson</LastEditorLname>` |
| <DisplayStartDate></DisplayStartDate> | The content block's start date. It is formatted as a string that represents Ektron CMS400's display of the date. |
| <FolderID></FolderID> | The ID of the folder in which the content block is contained.<br>For example:<br>`<FolderID>0</FolderID>` |
| <ContentStatus></ContentStatus> | The status of the content block. |
| <Language></Language> | The language of the content block.<br>For example:<br><Language>1033</Language> |
| <DisplayDateModified></DisplayDateModified> | edited. It is formatted as a string that represents Ektron CMS400's display of the date.<br>For example:<br>`<DisplayDateModified>1/19/2005 4:18:06 PM</DisplayDateModified>` |

| XML Tags | Description |
|---|---|
| <DisplayEndDate></DisplayEndDate> | The content block's end date. It is formatted as a string that represents Ektron CMS400's display of the date.<br>For example:<br>`<DisplayEndDate>12/31/9999 12:00:00 AM</DisplayEndDate>` |

Here is how the raw XML list summary output appears before an XSLT is applied.

```
<Content>
    <ID>24</ID>
    <Title>ASP.NET Unleashed</Title>
    <QuickLink>/CMS400Example/collection.aspx?id=24</QuickLink>
    <Teaser><p><em>ASP.NET Unleashed, Second Edition</em> is really big, really thorough, and
really <i>good</i>.</p></Teaser>
    <StartDate>1/1/0001 12:00:00 AM</StartDate>
    <DateModified>1/12/2005 7:53:31 PM</DateModified>
    <EndDate>12/31/9999 12:00:00 AM</EndDate>
    <LastEditorFname>Application</LastEditorFname>
    <LastEditorLname>Administrator</LastEditorLname>
    <DisplayStartDate></DisplayStartDate>
    <FolderID>0</FolderID>
    <ContentStatus></ContentStatus>
    <Language>0</Language>
    <DisplayDateModified>1/12/2005 7:53:31 PM</DisplayDateModified>
    <DisplayEndDate></DisplayEndDate>
</Content>
<Content>
    <ID>25</ID>
    <Title>Microsoft ASP.NET Programming with Microsoft Visual C#.NET</Title>
    <QuickLink>/CMS400Example/collection.aspx?id=25</QuickLink>
    <Teaser><p>Teach yourself how to write high-performance Web applications with ASP.NET and
Visual C# .NET--one step at a time.</p></Teaser>
    <StartDate>1/1/0001 12:00:00 AM</StartDate>
    <DateModified>1/4/2005 7:16:21 PM</DateModified>
    <EndDate>12/31/9999 12:00:00 AM</EndDate>
    <LastEditorFname>Application</LastEditorFname>
    <LastEditorLname>Administrator</LastEditorLname>
    <DisplayStartDate></DisplayStartDate>
    <FolderID>0</FolderID>
    <ContentStatus></ContentStatus>
    <Language>0</Language>
    <DisplayDateModified>1/4/2005 7:16:21 PM</DisplayDateModified>
    <DisplayEndDate></DisplayEndDate>
</Content>
```

# Archive List Summary Function

The Archive list summary function displays on a Web page a list of content summaries for archived content blocks. For example, you might have a current news page and an archived news page. Use the List Summary Function to display current news, and this function to display older news stories.

On the Content Block **schedule** screen, the user must set an **End Date** then choose the Archive and Display option to have the content blocks appear via this function.

Shown here is the format of the Archive List Summary function:

```
<% ecmListArchiveSummary    "StartingFolder",
                            Recursive,
                            ShowSummary,
                            "StyleInfo",
                            "OrderBy, Asc/Desc",
                            "ShowInfo",
                            MaxNumber,
                            "Spare1",
                            "Spare2"                %>
```

This function has the same parameters as the List Summary Function. See ”List Summary Function” on page 304 for details.

# Single Summary Function

The single summary tag is used to display the summary of one content block in the Ektron CMS400.NET Web site, instead of a list of all the summaries in a content folder. Seen below is the single summary tag:

```
<% ecmSingleSummary   "ContentID",
                      ShowSummary,
                      "StyleInfo",
                      "ShowInfo",
                      "Spare1"          %>
```

You can look at the single summary tag as just a simpler form of the List summary tag. The format for each of them is almost identical, but the single summary tag does not require as many attributes to be passed.

A list and description of the attributes are shown in this table:

| Attribute | Description | Required? |
|---|---|---|
| Content ID | This is the ID number of the content block the summary information will come from. | Yes |
| ShowSummary | Do you want the content summary to be displayed? | Yes |
| StyleInfo | Inline style information for the generated text hyperlink. | No |
| ShowInfo | Information that will appear to the right of the hyperlink. | No |
| Spare1 | Not yet implemented | MUST BE “” |

# Single Summary Example

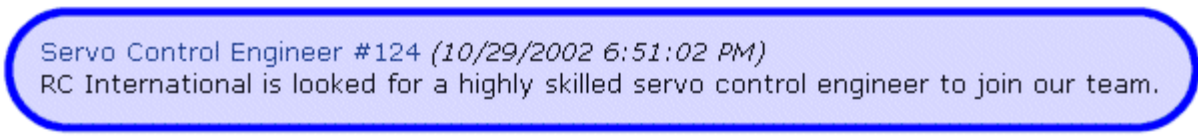
Shown below is an example of the single summary function in an ASP Web page.

```
<TR>
    <TD>
        <%  ecmSingleSummary "14",1,"","DateModified","" %>
    </TD>
</TR>
```

The previous example had the following properties:

- Displays the summary for the content block with an ID=14
- The summary will be shown
- No style was applied to the hyperlinks
- The last date the content was modified will be displayed

Here is how the summary will appear on the web page.


Servo Control Engineer #124 (10/29/2002 6:51:02 PM)
RC International is looked for a highly skilled servo control engineer to join our team.

# Single Summary Attributes

Listed below are all the different attributes and descriptions about them.

## Content ID

In this attribute, you will need to specify which content block you will want the summary to be displayed from.

## Show Summary

The show summary attribute will allow or disallow the summary to be displayed in the summary list. You have two options for this attribute:

| Value | Web Page View |
|---|---|
| 0 | Servo Control Engineer #124 |
| 1 | Servo Control Engineer #124<br>RC International is looked for a highly skilled servo control engineer to join our team. |

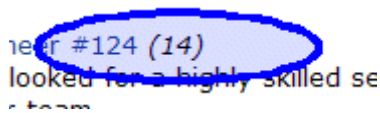
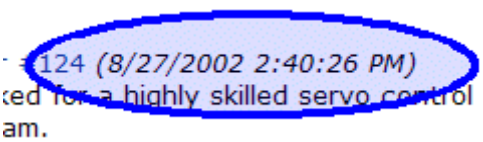
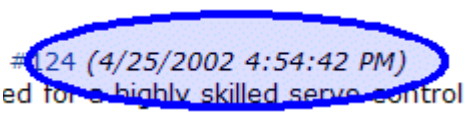
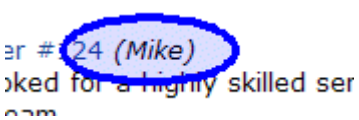
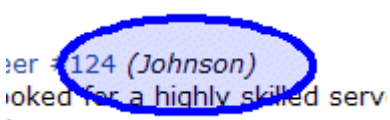
## Style Info

In this field, you may enter inline style information for the generated text hyperlinks displayed on the screen.

| Style Info | Web Page View |
|---|---|
| “” | Servo Control Engineer #124<br>RC International is looked for a highly skilled servo control engineer to join our team. |
| font-family:arial; font-weight:bold; background-color:#cccccc; border:solid blue 1pt; margin:2px; width:100%; | Servo Control Engineer #124<br>RC International is looked for a highly skilled servo control engineer to join our team. |

## Show Info

The show info field determines what sort of information is displayed to the right of the hyperlinked title. There are five different options that you may choose from:

| Value | Displays the... | Web Page View |
|---|---|---|
| ID | ID number for the content block. |  |
| DateModified | date the content was last modified. |  |
| DateCreated | date when the content was first created. |  |
| AuthorFname | first name of the user who last edited the content block. |  |
| AuthorLname | last name of the user who last edited the content block |  |

## Spare 1

There is one attribute fields at the end that are currently not used. This is here for future scalability. When you insert a single summary tag into your template, you must leave this attributes as ““

# RSS Summary Function

The RSS summary function is used to create an XML list of content summaries formatted for RSS Syndication.

Shown here is the format of the RSS summary function:

```
<% ecmRssSummary    "Folder",
                    Recursive,
                    "OrderBy, Asc/Desc",
                    MaxNumber,                %>
```

A list and description of the attributes are shown in this table:

| Attribute | Description | Options |
|---|---|---|
| Folder | This is the folder the summary information will come from. | Any folder |
| Recursive | Allow the tag to get summary info from the child folders. | 0 = Not Recursive<br>1 = Recursive |
| OrderBy, ASC/DESC | Order the hyperlinks by: | “Title”<br>“DateModified”<br>“DateCreated”<br>“Startdate”<br>“AuthorFname”<br>“AuthorLname” |
| | Order the hyperlinks in ascending or descending order. | “Title,asc”<br>“Title,desc” |
| MaxNumber | Maximum number of summaries returned. | 0 = Unlimited |

# RSS Summary Example

Shown below is an example of the ecmRssSummary function in an ASP Web page.

```
<TR>
    <TD>
        <% ecmRssSummary "\Books",0 "Title",0 %>
    </TD>
</TR>
```

The previous example has the following properties:

- Displays summaries from the \Books folder.
- Not recursive.
- The results will be displayed in ascending order by title.
- Unlimited results will be shown.

Here is how an RSS summary appears when viewed without a RSS reader on a web page.

```
<?xml version="1.0" ?>
<rss xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="2.0">
    <channel>
        <title>Books</title>
        <link>http://localhost/CMS400Example/WorkArea/WebServices/ContentWS.asmx</link>
        <description></description>
        <item>
            <title>Visual Basic.NET How to Program</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=23</link>
            <description>Visual Basic .NET How to Program, Second Edition is an awesome resource for the beginning VB.NET developer who is serious about the craft of programming. In 1,500 pages, this book covers an incredible amount of territory, from the evolution of programming languages and web development all the way to advanced data structures and web services.</description>
            <author>Application Administrator</author>
            <pubDate>3/3/2005 5:15:30 PM</pubDate>
        </item>
        <item>
            <title>Simply Visual Basic.NET 2003</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=26</link>
            <description>An Application-Driven Tutorial Approachp</description>
            <author>Application Administrator</author>
            <pubDate>1/8/2005 10:50:30 AM</pubDate>
        </item>
        <item>
            <title>Microsoft ASP.NET Programming with Microsoft Visual C#.NET</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=25</link>
            <description>Teach yourself how to write high-performance Web applications with ASP.NET and Visual C# .NET--one step at a time.</description>
            <author>Application Administrator</author>
            <pubDate>1/4/2005 7:16:21 PM</pubDate>
        </item>
        <item>
            <title>HTML for the World Wide Web with XHTML and CSS</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=27</link>
            <description>The easiest HTML book weve seen that still manages to be comprehensive, and to stay firmly grounded in the reality of todays demanding users and complex browsers.</description>
            <author>Application Administrator</author>
            <pubDate>1/19/2005 4:18:06 PM</pubDate>
        </item>
            </channel>
</rss>
```

# RSS Summary Attributes

Listed below are all the different attributes and descriptions about them.

## Starting Folder

In this attribute, you will need to specify which content folder you will want to get all the summaries from. There are two options for this attribute:

| Value | Description |
|---|---|
| “” | By leaving the value blank, all the summaries from the root content folder will be displayed. |
| “\Folder Path | You can specify a specific folder to display the summaries from by inserting the folder path |

## Recursive

When you setup the search to be recursive, you are allowing the search to search all the subfolders associated with the starting folder you had specified. You have two options for this attribute:

| Value | Description |
|---|---|
| 1 | Recursive - summaries in the starting folder’s subfolders will also be displayed. |
| 0 | Non-recursive - only summaries in the starting folder will be displayed. |

## Order By

This attribute will sort the summary results by different options. There are five different ways to sort the summaries:

| Value | Summaries sorted by |
|---|---|
| Title | title |

| Value | Summaries sorted by |
|---|---|
| DateModified | date modified |
| Date Created | date created |
| Startdate | The Start Date assigned to the content block when created. |
| AuthorFname | author's first names |
| AuthorLname | author's last name |

### *Ascending/Descending*

This attribute is an addition to the previous OrderBy attribute. By default, the list of the summaries are descending, meaning they will follow the path of A-B-C, or 1-2-3. When set to ascending, they will order by Z-Y-X or 3-2-1.You will have the option to re arrange these to display either in ascending or descending order.

To do this, add the following to the OrderBy attribute:

| Value | Description |
|---|---|
| asc, 1 | This will order the Summaries in an ascending order |
| desc, 0 | This will order the Summaries in a descending order. This is the default setting. |

By using the ascending/descending option, you will have the option to set the order of content blocks from most recent to oldest, or vice versa. As well as ordering the list by the user's first names or last names in ascending or descending order.

## Max Number

This attribute determines how many summaries appear on a page. Enter the number of summaries to be displayed. To display all summaries, enter 0 (zero) and you will display unlimited results.

# Collection Function

The collection function is used to display the list of links that were created as a collection in the Ektron CMS400.NET Workarea. The ecmCollection function is highly customizable, allowing you to easily define how the collection data appears on the Web page.

Shown here is the format of the ecmCollection function.

```
<% ecmCollection      "id",
                      "Display Function"     %>
```

The ecmCollection function attributes are explained below.

| Attribute | Description |
|---|---|
| id | The id of the collection that you would like to be displayed is defined here. |
| displayFunction | Specify a function that is defined in the API and determines how the appearance of the collection data on a Web page. The display function must also be defined in the ecmCollection function. |

*See Also:* The **Ektron CMS400.NET** User Manual Section "Working with Collections" > "Comparison of Collections, Menus, and the List Summary Features"

## Collection Examples

The following are examples of the collection function in the Ektron CMS400.NET sample Web site. One example uses the `ecmNavigation` display function, and the other uses the `ecmTeaser` display function.

The `ecmNavigation` and `ecmTeaser` display functions are sample display functions included with Ektron CMS400.NET. For

information about creating your own custom display functions, see "Creating Custom Display Functions" on page 334.

## The ecmNavigation Display Function

The following sample collection function is being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<% ecmCollection 1, "ecmNavigation" %>
```

In this example, the collection with an ID=1 is displayed, and the `ecmNavigation` function defines the appearance of the collection data on a Web page.

Here is how a collection might appear on a Web page.

RC International
Home | Products | Support | News | Careers | Search
Products Page
RC Cheetah
RC Lilly
RC Redstar
RC Sportster
Products
Welcome to RC International product page. Whether you are a
just a beginner, you will find fun and adventure in RC Internatio
RC Cars
RC Sportster — RC Internationals first entry
in to the competitive RC racing world was
the RC Sportster. The RC Sportster is
designed for the entry level RC enthusiast
RC Cheetah —
successful entr
racing world is
Cheetah is desi

Below is the source code for the `ecmNavigation` display function.

```
Function ecmNavigation(cInfo)

    dim html, info

    html = "<table border=""0"" cellspacing=""0"" cellpadding=""0""

                width=""100%"">"

    for each info in cInfo

        html = html & "<tr><td>  <a href="""

        html = html & info("ContentLinks")

        html = html & """>"

        html = html & Cstr(info("ContentTitle"))

        html = html & "</a></td></tr><tr><td> </td></tr>"

    next

    html = html & "</table>"

    ecmNavigation = html

end Function
```

As you can see, the `ecmNavigation` function is a simple ASP function that creates a table of the links to the specified collection. It displays the title of the content blocks by using the `info("ContentTitle"))` building block. See "Creating Custom Display Functions" on page 334 for more information.

## The ecmTeaser Display Function

The following is an example of a collection function being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<% ecmCollection 2, "ecmTeaser" %>
```

In this example, the collection with an ID=2 is displayed, and the `ecmTeaser` function defines the appearance of the collection data on the Web page. `ecmTeaser` displays the title and summary of each content block in the collection.

Here is how the collection might appear on a Web page.

RC International
Home | Products | Support | News | Careers | Search Login | Contact Ektron
Employment Opportunities
Plastic Molder #123 (8/27/2002 2:38:34 PM)
RC International is looking for an experienced plastics molder.
Servo Control Engineer #124 (8/27/2002 2:40:26 PM)
RC International is looked for a highly skilled servo control engineer to join our team.

The format of the collection on the Web page depends on the displayFunction that is used. Below is the source code for the display function `ecmTeaser`.

```
Function ecmTeaser(cInfo)

    dim html, info

    html = "<table border=""0"" cellspacing=""0"" cellpadding=""0""

                width=""100%"">"

    for each info in cInfo

        html = html & "<tr><td><a href="""

        html = html & CStr(info("ContentLinks"))

        html = html & """ >"

        html = html & Cstr(info("ContentTitle"))

        html = html & "</a> ("

        html = html & info("DateModified")

        html = html & ")</td></tr><tr><td>"

        html = html & Cstr(info("ContentTeaser"))

        html = html & "</td></tr><tr><td> </td></tr>"

    next

    html = html & "</table>"

    ecmTeaser = html

end Function
```

As you can see, the `ecmTeaser` function is an ASP function that creates a table of links to the specified collection. The `ecmTeaser` function displays the:

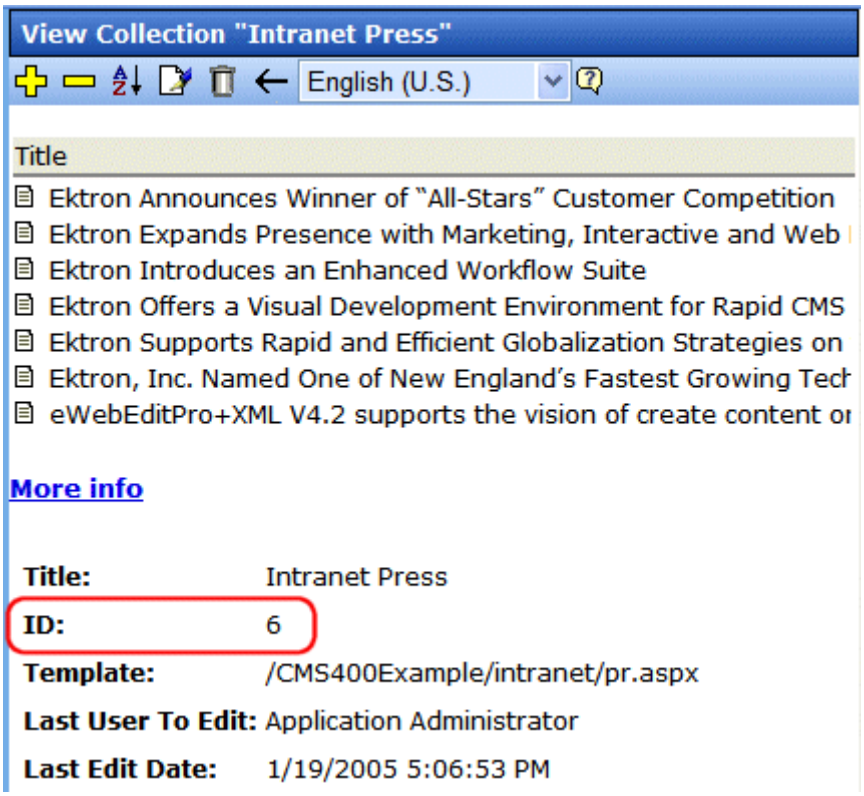
- Content title
- Date modified
- Content block summary

See ”Creating Custom Display Functions” on page 334 for more information.

# Collection Attributes

## ID

The collection ID number is assigned to the collection when it is created.


View Collection "Intranet Press"
English (U.S.)
Title
Ektron Announces Winner of "All-Stars" Customer Competition
Ektron Expands Presence with Marketing, Interactive and Web
Ektron Introduces an Enhanced Workflow Suite
Ektron Offers a Visual Development Environment for Rapid CMS
Ektron Supports Rapid and Efficient Globalization Strategies on
Ektron, Inc. Named One of New England's Fastest Growing Tech
eWebEditPro+XML V4.2 supports the vision of create content or
More info
Title: Intranet Press
ID: 6
Template: /CMS400Example/intranet/pr.aspx
Last User To Edit: Application Administrator
Last Edit Date: 1/19/2005 5:06:53 PM

## DisplayFunction

A display function determines which elements of the content blocks in a collection appear on a Web page. Via ASP scripting, you can create display functions or use any of the three included functions. You must define the display function in two places.

- In the ecmCollection function as a case
- In the API file as a function

# Creating Custom Display Functions

The following section explains how to create custom display functions for your ecmCollection function.

## Function Structure

All display functions use a similar structure:

```
Function yourCollectionDisplayFunction(cInfo)

    dim html, info

    html = html & "<b>List of Content Titles:</b><br>"

    for each info in cInfo

        html = html & info("ContentTitle") & "<br>"

    next

    yourCollectionDisplayFunction = html

end Function
```

### *Creating a Basic Display Function*

Explained below is a how to create a basic display function.

| | |
|---|---|
| 1. In the API file, define the display function. | `Function MyDisplayFunction(cInfo)`<br><br>`end Function` |
| 2. Declare any necessary attributes | `Function MyDisplayFunction(cInfo)`<br>`  dim html, info`<br><br>`end Function` |
| 3. Add any html or text to apply to the collection data as a whole, not individually (for example, add a opening and closing html table tag) | `Function MyDisplayFunction(cInfo)`<br>`  dim html, info`<br>`  html = "<ul>"`<br><br>`  html = html & "</ul>"`<br>`end Function` |

| | |
|---|---|
| 4. Add a `for` loop to cycle through each collection item, and apply the information defined | `Function MyDisplayFunction(cInfo)`<br>`  dim html, info`<br>`  html = "<ul>"`<br>`  for each info in cInfo`<br><br>`  next`<br>`  html = html & “</ul>”`<br>`end Function` |
| 5. Inside the `for` loop, create the string to be applied to each item generated by the collection data.<br><br>For information about the building blocks, see ”Building Blocks” on page 337. | `Function MyDisplayFunction(cInfo)`<br>`  dim html, info`<br>`  html = "<ul>"`<br>`  for each info in cInfo`<br><br>`  next`<br>`  html = html & “</ul>”`<br>`end Function` |
| This places an <li> tag at the beginning and end of each item . | `Function MyDisplayFunction(cInfo)`<br>`  dim html, info`<br>`  html = "<ul>"`<br>`  for each info in cInfo`<br>`  html = html & "<li>"`<br><br>`  html = html & "</li>"`<br>`  next`<br>`  html = html & “</ul>”`<br>`end Function` |
| This creates the hyperlink for the collection item. When the collection item is generated on the Web page, the hyperlink sends the visitor to the corresponding content block. | `Function MyDisplayFunction(cInfo)`<br>`  dim html, info`<br>`  html = "<ul>"`<br>`  for each info in cInfo`<br>`  html = html & "<li>"`<br>`  html = html & "<a href=" &`<br>`  info("ContentLinks") & ">" &`<br>`  Cstr(info("ContentTitle")) & "</a>"`<br><br>`  html = html & "</li>"`<br>`  next`<br>`  html = html & “</ul>”`<br>`end Function` |

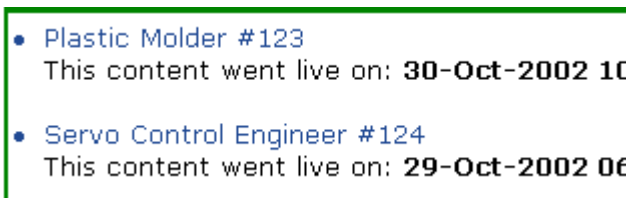
| | |
|---|---|
| This displays the summary of the content block in the collection. | `Function MyDisplayFunction(cInfo)`<br>`  dim html, info`<br>`  html = "<ul>"`<br>`  for each info in cInfo`<br>`  html = html & "<li>"`<br>`  html = html & "<a href=" &`<br>`  info("ContentLinks") & ">" &`<br>`  Cstr(info("ContentTitle")) & "</a>"`<br>`  html = html & Cstr(info("ContentTeaser"))`<br>`  html = html & "</li>"`<br>`  next`<br>`  html = html & “</ul>”`<br>`end Function` |
| 6. Add this line at the end of the function to return the HTML to be output to the ecmCollection function. | `Function MyDisplayFunction(cInfo)`<br>`  dim html, info`<br>`  html = "<ul>"`<br>`  for each info in cInfo`<br>`  html = html & "<li>"`<br>`  html = html & "<a href=" &`<br>`  info("ContentLinks") & ">" &`<br>`  Cstr(info("ContentTitle")) & "</a>"`<br>`  html = html & Cstr(info("ContentTeaser"))`<br>`  html = html & "</li>"`<br>`  next`<br>`  html = html & “</ul>”`<br>`  MyDisplayFunction = html`<br>`end Function` |

## Building Blocks

The sample display function created above uses only two building blocks. This section explains all building blocks you can use within a custom display function.

| Name | Description | Web Page View |
|---|---|---|
| `info("ContentTeaser")` | Displays content block summary | • Plastic Molder #123<br>RC International is looking for an experienced plastics molder.<br>• Servo Control Engineer #124<br>RC International is looked for a highly skilled servo control engineer to join our team. |
| `info("Comment")` | Displays content block comment | • Plastic Molder #123<br>This is the historical comment for Plastics M<br>• Servo Control Engineer #124<br>This is the historical comment for Servo Con |
| `info("ContentID")` | Displays content block ID number | • Plastic Molder #123<br>ID= 13<br>• Servo Control Engineer #124<br>ID= 14 |
| `info("ContentLinks")` | Displays content block path relative to your site's root | • Plastic Molder #123<br>/CMS400Sample/hr.asp?id=13<br>• Servo Control Engineer #124<br>/CMS400Sample/hr.asp?id=14 |

| Name | Description | Web Page View |
|---|---|---|
| `info("ContentTitle")` | Displays content block title. | • Plastic Molder #123<br>/CMS400Sample/hr.asp?id=13<br>• Servo Control Engineer #124<br>/CMS400Sample/hr.asp?id=14 |
| `info("DateModified")` | Displays date and time content block was last modified. | • Plastic Molder #123<br>8/27/2002 2:38:34 PM<br>• Servo Control Engineer #124<br>8/27/2002 2:40:26 PM |
| `info("DisplayDateModified")` | Displays date and time content block was last modified using the date/ time format specified in Setup.asp. | • Plastic Molder #123<br>27-Aug-2002 02:38:34 PM<br>• Servo Control Engineer #124<br>27-Aug-2002 02:40:26 PM |
| `info("EndDate")` | End date and time for the content block | |
| `info("DisplayEndDate")` | Displays end date and time for content block using date/time format specified in Setup.asp. | |
| `info("GoLive")` | Start date and time for content block | • Plastic Molder #123<br>This content went live on: **10/30/2002 10:**<br>• Servo Control Engineer #124<br>This content went live on: **10/29/2002 6:5** |

| Name | Description | Web Page View |
|---|---|---|
| `info("DisplayGoLive")` | Displays start date and time for the content block using the date/ time format specified in Setup.asp. |  |
| `info("EditorFname")` | First name of user who last edited content block. | |
| `info("EditorLname")` | Last name of user who last edited content block. | |

## Defining Display Functions

After you create a display function in the API, define it in the ecmCollection function as a new case.To do so, follow these steps.

1. Locate the ecmCollection function in the API file.
2. Find the line that contains `Select Case displayFunction`.
3. Under that line, insert the following, replacing `MyDisplayFunction` with the function you created.

```
Case "MyDisplayFunction"

    html = MyDisplayFunction(cInfo)
```

4. Save and close the file.

With the display function defined, and added to the ecmCollection function as a new case, you are ready to use the display function to define how collections look.

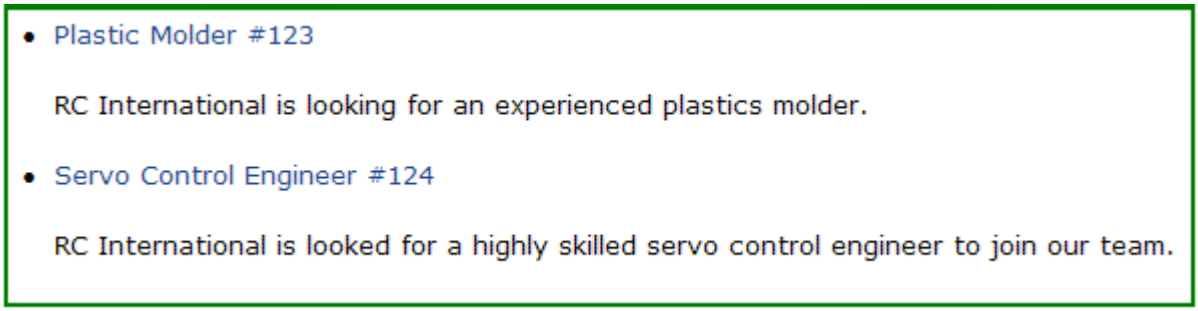
## Using a Custom Display Function

To use a custom display function, follow these steps.

1. Add the ecmCollection function to a template.
2. Specify the ID of the collection to be displayed.
3. Add the display function to define the collection's appearance.

```
<% ecmCollection "2", "MyDisplayFunction" %>
```

When displayed on the Web site, the collection looks like the following.


Plastic Molder #123
RC International is looking for an experienced plastics molder.
Servo Control Engineer #124
RC International is looked for a highly skilled servo control engineer to join our team.

# RSS Collection Function

The RSS collection function is used to create an XML list of links that were created as a collection in the Ektron CMS400.NET Workarea. The ecmRssCollection function is highly customizable, allowing you to easily define how the collection data appears on the Web page.

Shown here is the format of the ecmRssCollection function.

```
<% ecmRssCollection     “id”                    %>
```

The ecmRssCollection function attributes are explained below.

| Attribute | Description |
|---|---|
| id | The id of the collection that you would like to be displayed is defined here. |

*See Also:* The **Ektron CMS400.NET** User Manual Section “Working with Collections” > “Comparison of Collections, Menus, and the List Summary Features”

Here is how an RSS Collection appears when viewed without a RSS reader on a web page.

```
<?xml version="1.0" ?>
<rss xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="2.0">
    <channel>
        <title>Books</title>
        <link>http://localhost/CMS400Example/WorkArea/WebServices/ContentWS.asmx</link>
        <description></description>
        <item>
            <title>Visual Basic.NET How to Program</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=23</link>
            <description>Visual Basic .NET How to Program, Second Edition is an awesome resource for the beginning VB.NET developer who is serious about the craft of programming. In 1,500 pages, this book covers an incredible amount of territory, from the evolution of programming languages and web development all the way to advanced data structures and web services.</description>
            <author>Application Administrator</author>
            <pubDate>3/3/2005 5:15:30 PM</pubDate>
        </item>
        <item>
            <title>Simply Visual Basic.NET 2003</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=26</link>
            <description>An Application-Driven Tutorial Approachp</description>
            <author>Application Administrator</author>
            <pubDate>1/8/2005 10:50:30 AM</pubDate>
        </item>
        <item>
            <title>Microsoft ASP.NET Programming with Microsoft Visual C#.NET</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=25</link>
            <description>Teach yourself how to write high-performance Web applications with ASP.NET and Visual C# .NET--one step at a time.</description>
            <author>Application Administrator</author>
            <pubDate>1/4/2005 7:16:21 PM</pubDate>
        </item>
        <item>
            <title>HTML for the World Wide Web with XHTML and CSS</title>
            <link>http://localhost/CMS400Example/collection.aspx?id=27</link>
            <description>The easiest HTML book weve seen that still manages to be comprehensive, and to stay firmly grounded in the reality of todays demanding users and complex browsers.</description>
            <author>Application Administrator</author>
            <pubDate>1/19/2005 4:18:06 PM</pubDate>
        </item>
            </channel>
</rss>
```

# XML Collection Function

The XML collection function wraps XML around the parameters of a list of content that's created as a collection in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the collection information. The ecmCollectionXML function allows you to easily define how the collection data appears on the Web page.

Shown here is the format of the ecmCollectionXML function.

```
<% ecmCollectionXML    "id", "GetHtml"        %>
```

The ecmCollectionXML function attributes are explained below.

| Attribute | Description |
|---|---|
| id | The id of the collection that you would like to be displayed is defined here. |
| GetHtml | Determines whether the HTML contained within the content is included in the XML output.<br>0 = Do not get HTML<br>1 = Get HTML<br>Note: Ektron recommends using 0 (zero). HTML content can be large and this could cause display problems when your XML collection is displayed on your Web page. |

*See Also:* The **Ektron CMS400.NET** User Manual Section "Working with Collections" > "Comparison of Collections, Menus, and the List Summary Features"

When using the XML Collection function, the following tags are created around the properties of each item in the collection.

| XML Tags | Description |
|---|---|
| <Content></Content> | Declares a new section of content in the XML collection. |
| <ID></ID> | The content block ID.<br>For example:<br>`<ID>27</ID>` |
| <Title></Title> | The title of the content block.<br>For example:<br>`<Title>HTML for the World Wide Web with XHTML and CSS</Title>` |
| <QuickLink></QuickLink> | The quicklink associated with the content block.<br>For example:<br>`<QuickLink>/CMS400Example/collection.aspx?id=27</QuickLink>` |
| <Teaser></Teaser> | The summary of the content block.<br>For example:<br>`<Teaser>The easiest HTML book weve seen that still manages to be comprehensive.</Teaser>` |
| <Html></Html> | All of the HTML content contained in a content block. *See Also:* "GetHtml" on page 343<br>For example:<br>`<Html><em>ASP.NET Unleashed, Second Edition</em> is really big, really thorough, and really <i>good</i>. The explanations are simple and clear as anything weve seen in a .NET book. The sample code addresses virtually every facet of ASP.NET development, from little gems (code for validating credit cards) to full-fledged web stores. Best of all, Stephen Walther identifies the mistakes ASP.NET developers are most likely to make -- and presents detailed solutions. </Html>` |
| <StartDate></StartDate> | The content block's start date formatted as a .NET date type<br>For example:<br>`<StartDate>1/1/0001 12:00:00 AM</StartDate>` |

| XML Tags | Description |
|---|---|
| <DateModified></DateModified> | The last date the content block was modified.<br>For example:<br>`<DateModified>1/19/2005 4:18:06 PM</DateModified>` |
| <EndDate></EndDate> | The date the content block stops running on the Web site<br>For example:<br>`<EndDate>12/31/9999 12:00:00 AM</EndDate>` |
| <LastEditorFname></LastEditorFname> | The first name of the last person who edited the content block.<br>For example:<br>`<LastEditorFname>John</LastEditorFname>` |
| <LastEditorLname></LastEditorLname> | The last name of the last person who edited the content block.<br>For example:<br>`<LastEditorLname>Johnson</LastEditorLname>` |
| <Hyperlink></Hyperlink> | The hyperlink shown in the collection for linking to the content block.<br>For example:<br>`<Hyperlink><a href="/CMS400Example/collection.aspx?id=27">HTML for the World Wide Web with XHTML and CSS</a></Hyperlink>` |
| <DisplayStartDate></DisplayStartDate> | The content block's start date. It is formatted as a string that represents Ektron CMS400's display of the date. |
| <FolderID></FolderID> | The ID of the folder in which the content block is contained.<br>For example:<br>`<FolderID>0</FolderID>` |
| <ContentStatus></ContentStatus> | The status of the content block. |
| <Language></Language> | The language of the content block. |
| <DisplayDateModified></DisplayDateModified> | edited. It is formatted as a string that represents Ektron CMS400's display of the date.<br>For example:<br>`<DisplayDateModified>1/19/2005 4:18:06 PM</DisplayDateModified>` |

| XML Tags | Description |
|---|---|
| <DisplayEndDate></DisplayEndDate> | The content block's end date. It is formatted as a string that represents Ektron CMS400's display of the date.<br>For example:<br>`<DisplayEndDate>12/31/9999 12:00:00 AM</ DisplayEndDate>` |
| <EndDateAction></EndDateAction> | What should happen to the content block on its end date.<br>• archive display<br>• archive expire<br>• refresh report<br>For example:<br>`<EndDateAction>Archive_Expire</EndDateAction>` |
| <Comment></Comment> | Comments contained in the content block.<br>For example:<br>`<Comment>move picture below text</Comment>` |

Here is how the raw XML Collection output appears before an XSLT is applied.

```
<Content>
    <ID>24</ID>
    <Title>ASP.NET Unleashed</Title>
    <QuickLink>/CMS400Example/collection.aspx?id=24</QuickLink>
    <Teaser><p><em>ASP.NET Unleashed, Second Edition</em> is really big, really thorough, and
really <i>good</i>.</p></Teaser>
    <Html></Html>
    <StartDate>1/1/0001 12:00:00 AM</StartDate>
    <DateModified>1/12/2005 7:53:31 PM</DateModified>
    <EndDate>12/31/9999 12:00:00 AM</EndDate>
    <LastEditorFname>Application</LastEditorFname>
    <LastEditorLname>Administrator</LastEditorLname>
    <Hyperlink><a href="/CMS400Example/collection.aspx?id=24">ASP.NET Unleashed</a></
Hyperlink>
    <DisplayStartDate></DisplayStartDate>
    <FolderID>0</FolderID>
    <ContentStatus></ContentStatus>
    <Language>0</Language>
    <DisplayDateModified>1/12/2005 7:53:31 PM</DisplayDateModified>
    <DisplayEndDate></DisplayEndDate>
    <EndDateAction>Archive_Expire</EndDateAction>
    <Comment></Comment>
</Content>
<Content>
    <ID>25</ID>
    <Title>Microsoft ASP.NET Programming with Microsoft Visual C#.NET</Title>
    <QuickLink>/CMS400Example/collection.aspx?id=25</QuickLink>
    <Teaser><p>Teach yourself how to write high-performance Web applications with ASP.NET and
Visual C# .NET--one step at a time.</p></Teaser>
    <Html></Html>
    <StartDate>1/1/0001 12:00:00 AM</StartDate>
    <DateModified>1/4/2005 7:16:21 PM</DateModified>
    <EndDate>12/31/9999 12:00:00 AM</EndDate>
    <LastEditorFname>Application</LastEditorFname>
    <LastEditorLname>Administrator</LastEditorLname>
    <Hyperlink><a href="/CMS400Example/collection.aspx?id=25">Microsoft ASP.NET Programming
with Microsoft Visual C#.NET</a></Hyperlink>
    <DisplayStartDate></DisplayStartDate>
    <FolderID>0</FolderID>
    <ContentStatus></ContentStatus>
    <Language>0</Language>
    <DisplayDateModified>1/4/2005 7:16:21 PM</DisplayDateModified>
    <DisplayEndDate></DisplayEndDate>
    <EndDateAction>Archive_Expire</EndDateAction>
    <Comment></Comment>
</Content>
```

# Random Content

The random content function is used to randomly display the content of a content block that belongs in a specified collection.

Displayed here is the format for the random content function

```
<% ShowRandomContent “id”, XSLTId %>
```

Where the “**id**” represents the ID number of the collection the content blocks, that will be displayed, belong to.

And “**XSLTId**” is the ID number of the display XSLT that will be applied to the content blocks in the collections.

# Random Summary

Similar to the random content function, the random summary function randomly displays a content block summary that belongs in a specified collection.

Here is the format for the random content function

```
<% ShowRandomSummary “id” %>
```

The “id” represents ID number of the collection to which the content blocks belong.

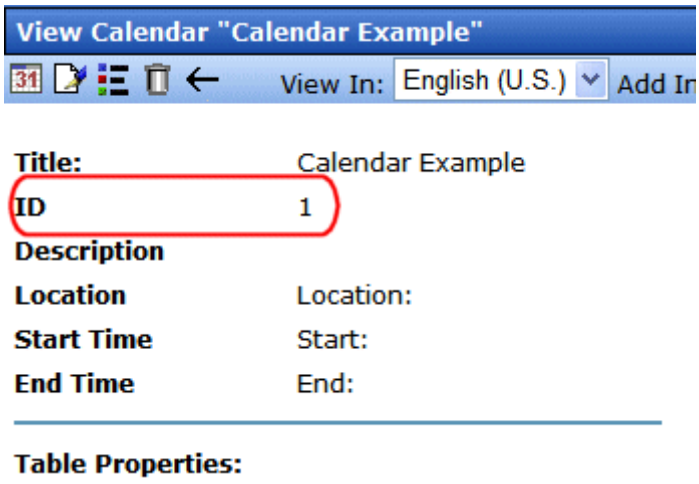
# Event Calendar Function

The ecmEvtCalendar function displays a month view event calendar that has been created in Ektron CMS400.NET. Here is the format for the event calendar custom function.

```
<% ecmEvtCalendar “id” %>
```

**NOTE** You can use ecmCalendar to call a calendar and display a different view, such as daily. For more Information, see ”Calendar Function” on page 353

The **“id”** argument represents the ID number of the event calendar in the Workarea.




## Event Calendar Style Sheet

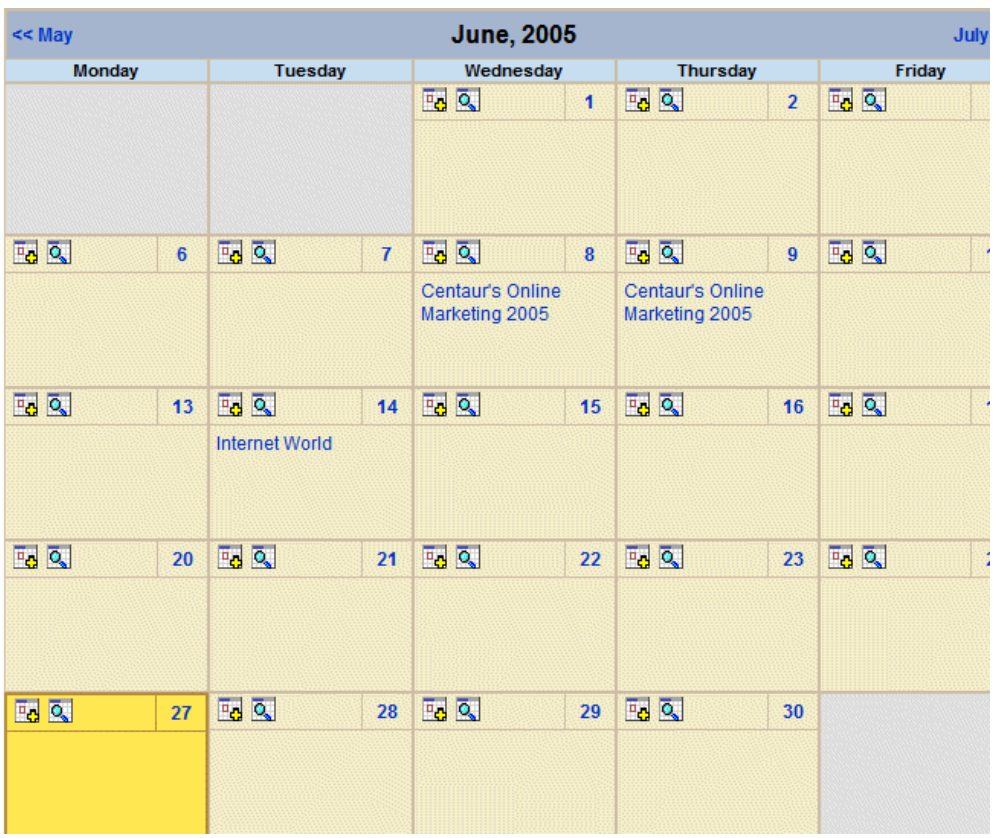
Before you insert an event Calendar into your Ektron CMS400.NET template, you should make a reference to the style sheet, `\CMS400example\calendarStyles.css`, in your template. This style sheet contains styles that define the Calendar.

# Event Calendar Sample

Here is an example of an event Calendar in an Ektron CMS400.NET template.

```
<tr>
    <td>
        <% ecmEvtCalendar "1" %>
    </td>
</tr>
```

When this code is displayed in a browser, the Calendar with an ID-1 is displayed. Here is the calendar displayed on a Web Page

| If... | Calendar Display |
|---|---|
| Logged in |  |

| If... | Calendar Display |
|---|---|
| Logged Out | |



As you can see, when you are logged into the Web site, you can add and view Calendar events. When logged out, you can only view calendar events.

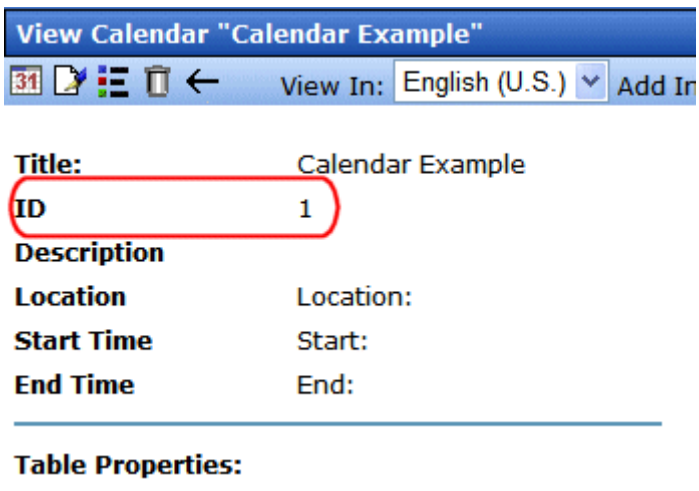
# Calendar Function

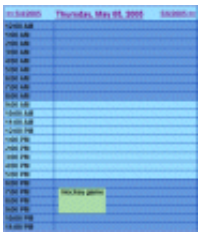
The ecmCalendar function displays a calendar that has been created in Ektron CMS400.NET. Here is the format for the event calendar custom function.

```
<% ecmCalendar “id”, “DisplayType” %>
```

**NOTE** While you can display a month view calendar with ecmCalendar, you can also use ecmEvtCalendar to display a month view calendar. For more Information, see ”Event Calendar Function” on page 350

The ecmCalendar function attributes are explained below.

| Attribute | Description |
|---|---|
| id | The **“id”** argument represents the ID number of the event calendar in the Workarea.<br>  |

| Attribute | Description |
|---|---|
| DisplayType | Specify the Calendar display type.<br>• **Month** - A 30 day calendar month view.<br> <br>• **Day** - Single day view.<br>  <br>• **Inline** - Display events that occur from the start date of the calendar through the end date.<br> <br>• **MonthUpWithEvent** - Displays a month calendar. Events appear to the right of the calendar when you click on a day.<br> |

# Calendar Style Sheet

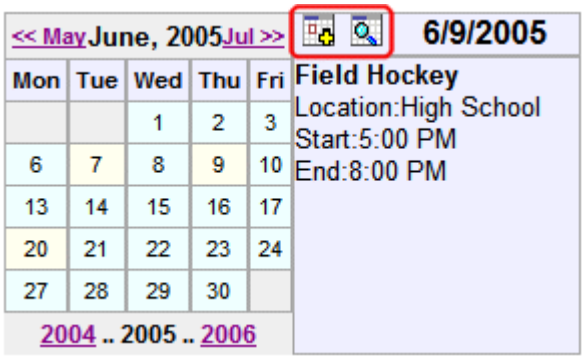
Before you insert an event Calendar into your Ektron CMS400.NET template, you should make a reference to the style sheet, `\CMS400example\calendarStyles.css`, in your template. This style sheet contains styles that define the Calendar.

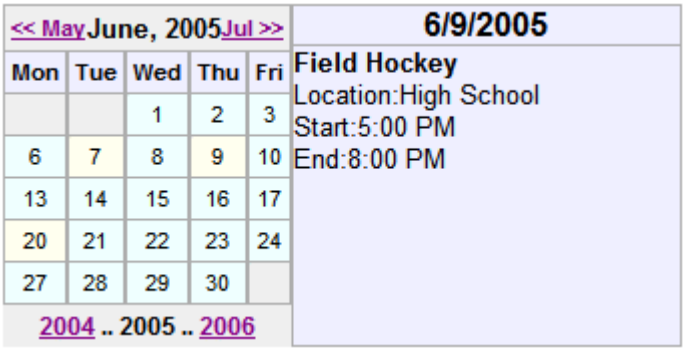
# Event Calendar Sample

Here is an example of an event Calendar in an Ektron CMS400.NET template.

```
<tr>
    <td>
        <% ecmCalendar "1", “MonthUpWithEvent” %>
    </td>
</tr>
```

When this code is displayed in a browser, the Calendar with an ID of 1 and DisplayType of MonthUpWithEvent is displayed. Here is the calendar displayed on a Web Page

| If... | Calendar Display |
|---|---|
| Logged in | Notice the add event and view event buttons that appear when logged in.  |

| If... | Calendar Display |
|---|---|
| Logged Out |  |



As you can see, when you are logged into the Web site, you can add and view Calendar events. When logged out, you can only view calendar events.

# Form Function

The ecmFormBlock and ecmFormBlockWithFormTag functions display an HTML form content block. Both functions have the same parameters and display the form the same way. The only difference is ecmFormBlockWithFormTag automatically inserts form tags, ecmFormBlock does not.

The following is an example of form tags.

```
<form name="myform" action="#" onsubmit="return
EkFmValidate(this);" method="post">

</form>
```

When using ecmFormBlock, form tags need to be added manually.

For more information about HTML forms, see the **Ektron CMS400.NET** Administrator manual chapter “Working with HTML Forms.”

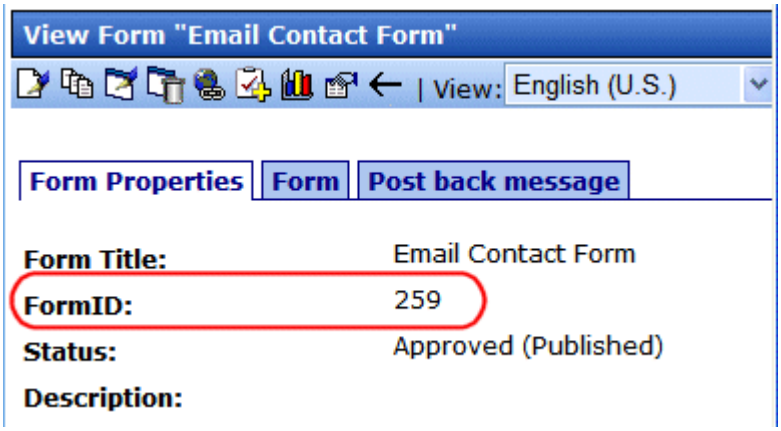
**IMPORTANT!** If you create a template for an existing form content block, you must manually change its quicklink to point to the new template. This change does not occur automatically. This procedure is described in the Ektron CMS400.NET Administrator manual chapter “Managing Library Assets”, section “Updating Default Template for Multiple Quicklinks.”

Here are the two formats for the form custom function.

**<% ecmFormBlock “id” %>**
**or**
**<% ecmFormBlockWithFormTag “id” %>**

**NOTE** As an alternative, you can use the multipurpose function, which can display either a content block or a content block associated with a form. For more information, see ”MultiPurpose Function” on page 360.

The “id” argument represents the form’s ID number, visible when viewing the form from the View Form screen.




Here is an example of a form content block in an Ektron CMS400.NET template.

```
<tr>
    <td>
        <% ecmFormBlockWithFormTag "367" %>
    </td>
</tr>
```

When this code is read by a browser, the following is displayed.

When a visitor to your site submits a form, the form uses the form information to process the data.

# MultiPurpose Function

The ecmMultiPurpose function displays either a content block or an content block associated with a form. As a result, you can use one template to display both types of content block.

Here is the format for the Multipurpose function:

```
<% ecmMultiPurpose “id” %>
```

To pass in a standard content block, pass the id value in the url parameter, as illustrated below.

```
http://localhost/cms400sample/index.asp?id=2
```

To pass in a Form content block, pass the ekfrm value in the url parameter, as illustrated below.

```
http://localhost/cms400sample/index.asp?ekfrm=2
```

**NOTE** For more information about passing the id value as a url parameter, see ”Dynamic Content Block” on page 283.

If the id value or ekfrm value does not exist, a blank template appears.

If the url parameter is not passed in or recognized, the function calls ecmContentBlock and passes the DefaultID value as the id argument. If the DefaultID value is zero (0), ecmContentBlock is not called.

# Metadata Function

The metadata tag is used to display the metadata for one or more content blocks in the Ektron CMS400.NET Web site. This custom tag lets you generate the list of metadata edited by the authors and insert it in the source of the Web page. Below is an example of the metadata custom tag.

```
<%  ecmMetaData "ItemList","Spare1" %>
```

The attributes are described below.

| Attribute | Description | Required? |
|---|---|---|
| ItemList | List of the content block ID numbers, and exclusions *See Also:* "Creating an Item List" on page 361 and "Item List" on page 366 | No |
| Spare1 | Not yet implemented *See Also:* "Spare 1" on page 368 | MUST BE "" |

## Creating an Item List

The Item List for the ecmMetadata tag has the following format.

```
"[content block id; Exclusion list]...[content block id; Exclusion list]"
```

| Attribute | Description | Required? |
|---|---|---|
| [ (left bracket) | Delineates the beginning of a content block id/exclusion list pair | Yes |
| Content Block ID | The ID of the content block whose metadata you want to retrieve | Yes |

| Attribute | Description | Required? |
|---|---|---|
| ; (semicolon) | Separator: Separates content block id from the exclusion list | No (**Yes** if including an **exclusion** list) |
| Exclusion List | A comma delimited list of metadata you do not want to be retrieved for this content block. | No |
| ] (right bracket) | Delineates the end of a content block id/exclusion list pair | Yes |

# ecmMetadata Example

For this example, our administrator has defined the following metadata types for Ektron CMS400.NET.

| Name | Style | Type | Editable | Required | Remove Dups | Case Sensitive |
|---|---|---|---|---|---|---|
| Title | n/a | HTML | Yes | Yes | No | n/a |
| Keywords | Name | Meta | Yes | Yes | Yes | No |
| Content-Type | http-equiv | Meta | No | Yes | Yes | n/a |

Our example Web page has two content blocks displayed.

- content block 1, the company's home page introduction text
- content block 2, a sidebar containing the lead story for the week

The metadata entered by your content editors for these two content blocks are:

**Content Block 1:**
Title – "Welcome to Our Home Page"
Keywords – "Home; page; company; Welcome"
Content-Type -- "text/html; charset=iso-8859-1"

**Content Block 2:**
Title – “Announcing our New Product!”
Keywords – “Announce; company; product; widget”
Content-Type -- “text/html; charset=iso-8859-1“

To include both content blocks metadata in a Web page, the ecmMetadata call looks like this.

```
<html>
    <head>
        <% ecmMetadata “[1][2;Title,Content-Type]”, “” %>
        (rest of head)
    </head>
    <body>
        (rest of web page document)
    </body>
</html>
```

In the above example, the ecmMetadata call is contained in the `<head>` of the HTML document (since all HTML metadata is required to exist there). When the Web page is viewed, the Web page source contains the following information:

```
<html>
    <head>
        <Title>Welcome to Our Home Page</Title>
        <meta name=”Keywords” content= Home; page; company; Welcome; Announce;
        product; widget”>
        <meta http-equiv=”Content-Type” content=”text/html; charset=iso-8859-1”>
        (rest of head)
    </head>
    <body>
        (rest of web page document)
```

```
    </body>

</html>
```

The ecmMetadata constructed three HTML tags based on the administrator defined parameters and the ecmMetadata call.

The metadata tag was constructed in the page because:

- **[1]** - Displays all metadata content for the content block ID=1. In this example, the title, keywords, and content-type were defined, so they were displayed as:

```
<Title>Welcome to Our Home Page</Title>

    <meta name="Keywords" content= Home; page; company; Welcome;

        Announce; product; widget">

    <meta http-equiv="Content-Type" content="text/html;

        charset=iso-8859-1">
```

- **[2;title,content-type]** - Displays all metadata content for content block ID=2, *except* for its title and content-type definitions.

```
<Title>Welcome to Our Home Page</Title>

    <meta name="Keywords" content= Home; page; company; Welcome;

        Announce;  product; widget">

    <meta http-equiv="Content-Type" content="text/html;

        charset=iso-8859-1">
```

The next section provides more examples.

# More ecmMetadata Examples

Shown below is an example of the metadata function in an ASP Web page.

```
<TR>

    <TD>

        <%  ecmMetaData "[1][2;title][14;keywords]","" %>
```

```
    </TD>
</TR>
```

The previous example has the following properties:

- All metadata for content block ID=1 is inserted into the page
- All metadata for the content block ID=2 except for its title is inserted into the page
- All metadata for the content block ID=14 except for its keywords is inserted into the page

### *Using a Dynamic Metadata Tag*

You can use a metadata tag in a template to define the metadata for a specific content block. However, you can also use a dynamic metadata tag combined with a content block tag to show the metadata for all content blocks on a Web page.

This allows you to create a template and not have to keep updating the tag by adding the new ID and exclusion list. Shown below is the sample code for this.

```
<!-- #include virtual="/CMS400.NETsample/CMS400.NETscripts/ektronAPI.asp" -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
    <head>
        <% Dim id
            if (request.QueryString("id")<> "") then
                id = request.QueryString("id")
            else
                id = 1
            end if
                ecmMetadata "[" & id & "]", ""
        %>
    </head>
    <body>
```

```
            <% ecmContentBlock(request.QueryString("id")) %>

        </body>

    </html>
```

In this example, if you specify a content block in the URL and the page is displayed, the metadata takes that ID and displays the proper metadata in the source for the content block.

# ecmMetadata Attributes

Below are the attributes and their descriptions.

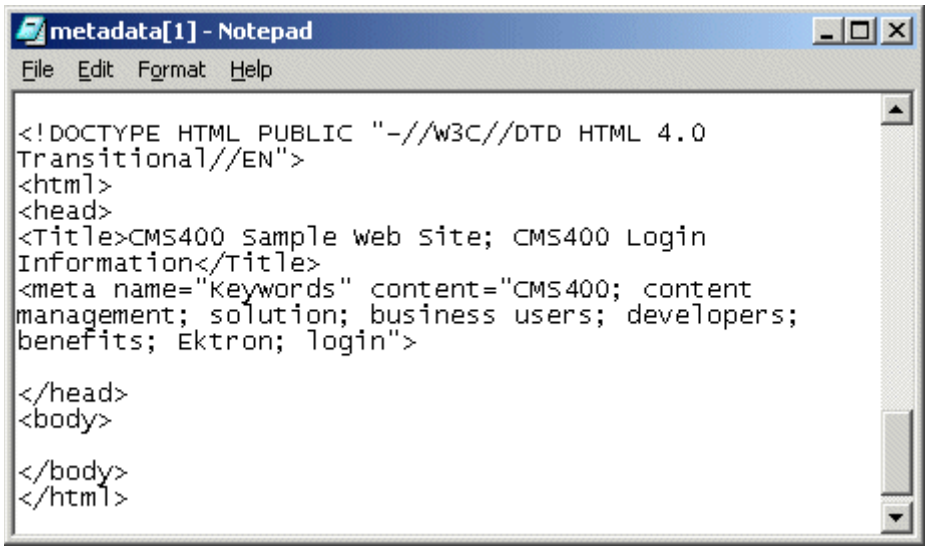
## Item List

In this attribute of the ecmMetadata tag, you need to specify two values.

| Value | Specifies |
|---|---|
| ID | The content block ID whose metadata will appear |
| Exclusion List | The metadata to be excluded. *See Also:* ”Creating an Item List” on page 361 |

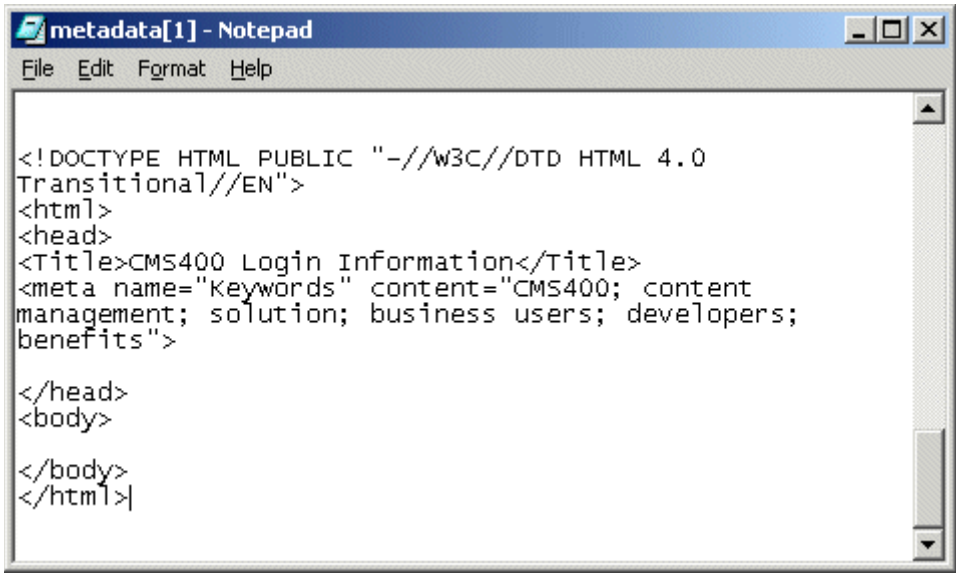
Some examples might be:

```
<%  ecmMetaData "[1][2]","" %>
```



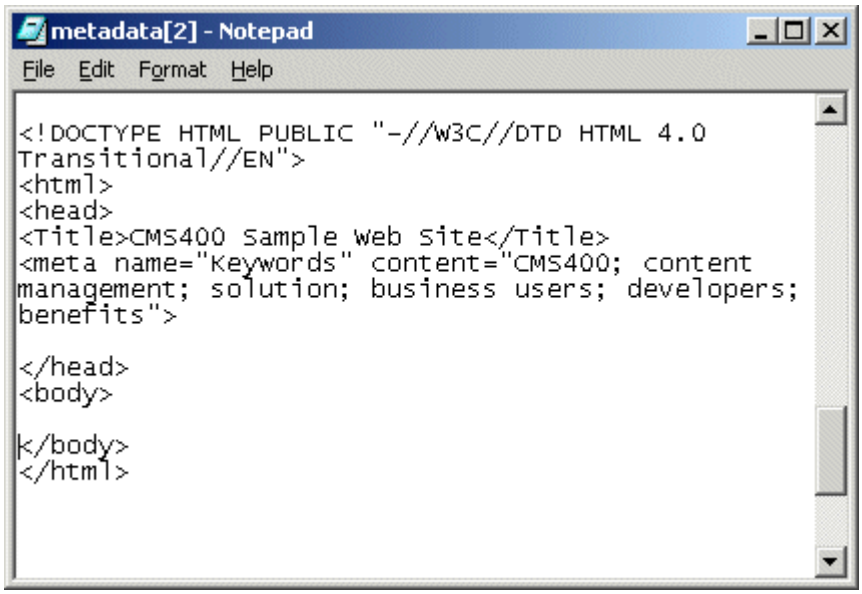

In this example, all metadata from content block ID=1 and 2 is displayed.

```
<%  ecmMetaData "[1;title][2;keywords]","" %>
```




In this example, all the metadata from content block ID=1 will be displayed *except* for the title. Also, all the metadata for content block ID=2 is displayed *except* its keywords.

```
<%  ecmMetaData "[1][2;title,keywords]","" %>
```




In this example, all metadata from content block ID=1 is displayed, and all metadata from content block ID=2 is displayed *except* its title and keywords.

## Spare 1

One attribute field at the end is currently not used. It is for future scalability. When you insert a single metadata tag into a template, you must leave this attributes as “”.

# Show Site Language Function

This function lets a visitor to your Web site select a language in which to view the site. Here is what the function looks like when published on a Web page.

It lists all languages selected on the **Settings > Configuration > Language settings** screen. (For more information, see the “Multi-Language Support “chapter of the **Ektron CMS400.NET** Administrator manual.)

Below is an example of the Show Site Language custom tag.

```
<% ecmShowSiteLanguage(FieldName)%>
```

You can place this function in any location of any page on your site. By default, it appears in the lower right corner of the home page.

**NOTE** This function only works with Asp and ASP.Net.

The attribute is described in this table.

| Attribute | Description |
|---|---|
| FieldName | An optional argument. You can leave it blank and allow the drop-down list to be populated by the available languages for your Web site. For example:<br>`<%=ecmShowSiteLanguage(‘’)%>`<br>Or you can pass in an argument such as:<br>`<%=ecmShowSiteLanguage(‘mysitelanguage’)%>` |

# DHTML Menus

**Ektron CMS400.NET**'s Menu feature lets users create and maintain a dropdown menu system for your Web site. The Menu feature lets you display menu options that link to content blocks, library assets, external hyperlinks and submenus.

To learn more about menus, please refer to the "Menu Feature" chapter of the **Ektron CMS400.NET** User Manual.

## Additional Documentation in Ektron Knowledge Base

This chapter explains how to implement standard menus that ship with **Ektron CMS400.NET**. Ektron's Web site also provides Knowledge Base articles that explain how to extend the standard menu features.

## Displaying a Menu on a Web Page

### GenerateMenu and DropMenu

**IMPORTANT!** The menu functions described below are only one example of how to implement menus. The scripts reside in the Application API file. You can implement your own menu system using JavaScript.

Use the `GenerateNextGenMenu` and `DropNextGenMenu` custom functions to display a DHTML menu on a Web page. Place `GenerateNextGenMenu` once on the page to provide an infrastructure for `DropNextGenMenu`. Then, place `DropNextGenMenu` as needed to implement individual menus.

Follow these steps to display a menu on a Web template.

1. Place the following code directly after the page's <body> tag.

```
<%=(ecmDHTML_GenerateNextGenMenu("menu id number"))%>
```

   For example:

```
<%=(ecmDHTML_GenerateNextGenMenu("1"))%>
```

   To display more than one menu, separate each id number with a comma. For example:

```
<%=(ecmDHTML_GenerateNextGenMenu("1,2,3"))%>
```

2. If the user clicks *text* to display the menu, place the following code within the template file where the text would appear.

```
<%= ecmDHTML_DropNextGenMenu(menu id number,"text user clicks to display menu")%>
```

   For example:

```
<%= ecmDHTML_DropNextGenMenu(1,"Products")%>
```

   If the user clicks *a graphic* to invoke the menu, use this syntax:

```
<%= ecmDHTML_DropNextGenMenu(menu id number,'<img src=graphic file name>')%>
```

   For example:

```
<%= ecmDHTML_DropNextGenMenu(1,'<img src=mymenugraphic.jpg>')%>
```

### Displaying Menus in a Multi-language System

If your Web site supports multiple languages, and the user selects a language before viewing the menu, if an edition of the menu is available in that language, it appears. If not, nothing appears.

For more information, see the **Ektron CMS400.NET** Administrator Manual section "Multi-Language Support "> "Working with Multi-Language Content" > "Working with Menus in a Multi-Language System."

## Customizing the Menu's Appearance

You can customize a menu's appearance, such as the font style, color, and size. So, you have the flexibility to coordinate a menu with the design your Web site.

To customize a menu in **Ektron CMS400.NET**. You can change:

- ”ek_Menu.js” on page 372

# ek_Menu.js

You can change the look and feel of a menu by changing the ek_menu.js file. If you used the default settings when installing **Ektron CMS400.NET**, this file is in *webroot*`/CMS400Example/Workarea/java/`.

Use the following table to help you change menu attributes.

| Attribute name | Specifies the | Default value |
|---|---|---|
| fontSize | Size of menu text | 14 |
| fontWeight | Thickness of menu text<br>You can set FontWeight by name (`Bold` or `Plain`) or value. Values range from 0 -1000. | Plain |
| fontFamily | Font style of menu text<br>*See Also:* ”FontFamily” on page 373 | arial,helvetica,espy, sans-serif |
| fontColor | Color of menu text<br>Note: When changing the color, you can specify the color name or code. | #000000 |
| fontColorHiLite | Color of text when menu item is highlighted | #ffffff |
| bgColor | Color of border between menu items | #555555 |
| menuBorder | Width (in pixels) of border around main menu | 1 |
| menuItemBorder | Width (in pixels) of border around each menu item | 1 |
| menuItemBgColor | Background color of menu items | #6699ff |
| menuLiteBgColor | Color of main menu border highlight<br>*See Also:* ”menuLiteBgColor” on page 374 | #ffffff |

| Attribute name | Specifies the | Default value |
|---|---|---|
| menuBorderBgColor | Color of the main menu border | #6699ff |
| menuHiLiteBgColor | Background color of highlighted menu item | #000084 |
| menuContainerBgColor | Background color of the *container*<br>Note: The container is like a  tag that displays the menu. When you set the container color, it appears at the top and left side of the menu frame only. | #6699ff |
| childMenuIcon | Path to the submenu icon (the image used to indicate that a submenu is available from this menu item).<br> <br><br>To display a different image, either provide an external path to it or upload the image to **Ektron CMS400.NET** then provide a local path. | `./Workarea/images/application/arrows.gif` |
| childMenuIconHiLite | The path to the icon that appears when you hover the mouse over a menu item that leads to a submenu.<br>To display a different image, either provide an external path to it or upload the image to **Ektron CMS400.NET** then provide a local path. | `./Workarea/images/application/arrows2.gif` |

## FontFamily

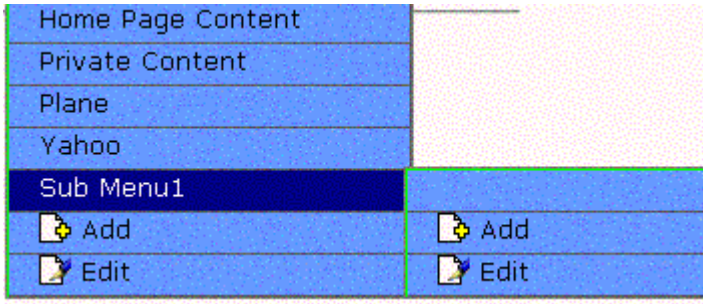
Use the fontFamily attribute to specify the menu font. It must be supported by the browser.

You can assign to this property several, comma-separated values.

**NOTE** To display a font family other than the browser default, specify only the font name and no other comma-separated values. If you specify more than one value for the font family, Ektron CMS400.NET uses the default IE settings.

## menuLiteBgColor

### Example



# XML Menu

The ecmGetMenuXML function wraps XML around the properties of a menu created in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the menu. The ecmGetMenuXML function allows you to easily define how the menu data appears on the Web page.

Shown here is the format of the ecmGetMenuXML function.

```
<% ecmGetMenuXML (id) %>
```

The ecmGetMenuXml function attributes are explained below.

| Attribute | Description |
|---|---|
| Id | The id that associated with the menu. |

Here is how the raw XML Menu output appears before an XSLT is applied.

```
<Item>
    <ItemID>1</ItemID>
    <ItemType>Submenu</ItemType>
    <ItemTitle>MenuExample</ItemTitle>
    <ItemImage></ItemImage>
    <ItemImageOverride>False</ItemImageOverride>
    <Menu>
        <ID>1</ID>
        <Title>MenuExample</Title>
        <Template></Template>
        <Type>content</Type>
        <Link></Link>
        <ParentID>0</ParentID>
        <AncestorID>1</AncestorID>
        <FolderID>0</FolderID>
        <Description></Description>
        <ImageOverride>False</ImageOverride>
        </Item>
        <Item>
            <ItemID>2</ItemID>
            <ItemType>Submenu</ItemType>
            <ItemTitle>Book</ItemTitle>
            <ItemImage></ItemImage>
            <ItemImageOverride>False</ItemImageOverride>
            <Menu>
                <ID>2</ID>
                <Title>Book</Title>
                <Template></Template>
                <Type>content</Type>
                <ParentID>1</ParentID>
                <ImageOverride>False</ImageOverride>
                <Item>
                    <ItemID>23</ItemID>
                    <ItemType>content</ItemType>
                    <ItemTitle>VB.NET How to</ItemTitle>
                    <ItemImage></ItemImage>
                    <ItemLink>/CMS400Example/collection.aspx?id=23</ItemLink>
                    <ItemTarget>2</ItemTarget>
                </Item>
            </Menu>
        </Item>
    </Menu>
</Item>
```

# Custom ColdFusion Functions

Ektron CMS400.NET maintains and display content through the use of custom Ektron ColdFusion functions and ComObjects.

When using custom ColdFusion functions, the following paths must be updated in the application.cfm.

```
<cfset application.WSDLpath = "http://localhost/CMS400Example/WorkArea/WebServices/
ContentWS.asmx?WSDL" />
```

and

```
<cfset application.AppPath = "/CMS400Example/WorkArea/platforms/ColdFusion/" />
```

The `application.WSDLpath` needs to updated with the location of the WSDL file. For example, For example, if your site is `www.myektronsite.com`, the WSDLpath would be `http://www.myektronsite.com/WorkArea/WebServices/ContentWS.asmx`.

The application.AppPath needs to updated with the path to the ColdFusion custom Functions.

To customize you Ektron CMS400.NET application, you will need to be familiar with these tags

| Function | Description | More Information |
|---|---|---|
| **Calendar Functions** | | |
| Event Calendar | Displays a month view event calendar | ”Event Calendar Function” on page 394 |
| **Collection Functions** | | |
| Collection | Displays links generated by collections | ”Collection Function” on page 388 |

| Function | Description | More Information |
|---|---|---|
| CollectionXML | Creates XML for a collection by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed. | "XML Collection Function" on page 393 |
| Random Content | Randomly displays the content of a collection item | "Random Content Function" on page 381 |
| Random Summary | Randomly displays the summary of a collection item | "Random Summary Function" on page 382 |
| RSS Collection | Creates an RSS feed out of the items in a collection. | "RSS Collection Function" on page 394 |
| **Content Functions** | | |
| Content Block | Displays and manages a content block | "Content Block Function" on page 380 |
| Form | Displays a content block associated with a form | "Form Function" on page 397 |
| Multipurpose | Displays either a content block or a form | "MultiPurpose Function" on page 398 |
| **Language Functions** | | |
| Language Select | Lets a site visitor select a language in which to view site. | "Language Select Function" on page 403 |
| **List Summary Functions** | | |
| Archive ListSummary | Displays summaries of archived content blocks | "Archive List Summary Function" on page 387 |
| List Summary | Displays a list of content block summaries | "List Summary Function" on page 384 |
| List Summary XML | Creates XML for a list summary by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed. | "XML List Summary Function" on page 385 |

| Function | Description | More Information |
|---|---|---|
| RSS List Summary | Creates an RSS feed out of the items in a list summary. | "RSS List Summary Function" on page 386 |
| Single Summary | Displays the summary for a content block. | "Single Summary Function" on page 387 |
| **Login Functions** | | |
| Login | Adds the login or logout button | "Login Function" on page 379 |
| **Menu Functions** | | |
| DropHTMLMenu | Used to display a DHTML menu | "DHTML Menus" on page 401 |
| GenerateMenu | Works with DropHTML menu to display DHTML menus | "DHTML Menus" on page 401 |
| XML Menu | Creates XML for a menu by wrapping tags around all of the menu items properties. | "XML Menu Function" on page 402 |
| **Metadata Functions** | | |
| Metadata | Displays the metadata for the current Web page | "Metadata Function" on page 398 |
| **Search Functions** | | |
| Search | Inserts a text box with a search button | "Search Function" on page 382 |
| Search Display | Creates an area to display search results | "Search Display Function" on page 383 |

# Login Function

This tag is responsible for displaying the following buttons on the Web page.

This function is basically the same as the Login function in ASP. See ”Login Function” on page 46 for details. The only difference is the syntax used to implement the function. This is listed below.

### Login Tag Example

```
<tr>

    <td>

        <cfmodule template="#request.ecm.AppPath#ecmLogin.cfm">

    </td>

</tr>
```

# Content Block Function

The content block and content blockEX tags are used to display a content block on an Ektron CMS400.NET Web page.

Shown here is the format for the content block function:

```
<CFMODULE template="#request.ecm.AppPath#ecmContentBlock.cfm"

          ID=""

>
```

Shown here is the format for the content blockEX function:

```
<CFMODULE template="#request.ecm.AppPath#ecmContentBlockEx.cfm"

          ID=""

          XSLT=""

          Override=""

>
```

This functions are basically the same as the ContentBlock and ContenBlockEX function in ASP. See ”Content Block Function” on page 49 for details.

The only differences between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion examples are below.

### Static Content Block

```
<tr>

   <td>

      <CFMODULE id="1" template="#request.ecm.AppPath#ecmContentBlock.cfm">

    </td>

</tr>
```

### Dynamic Content Block

```
<tr>

  <td>

      <CFMODULE id="#id#" template="#request.ecm.AppPath#ecmContentBlock.cfm">

  </td>

</tr>
```

### XML Content Block

```
  <tr>

    <td>

      <CFMODULE id="13" XSLT= “” override= “1”

      template="#request.ecm.AppPath#ecmContentBlockEx.cfm">

    </td>

  </tr>
```

## Random Content Function

The random content function is used to randomly display the content of a content block that belongs to a specified collection. Below is the format for the random content function:

```
<CFMODULE template="#request.ecm.AppPath#ecmShowRandomContent.cfm"

                ID=""

                XSLT=""

                Override=""

      >
```

This function is basically the same as the Random Content function in ASP. See "Random Content" on page 121 for details.

# Random Summary Function

SImilar to the random content function, the random summary function can randomly display a content block summary that belongs to a specified collection. Here is the format for the random content function

```
<CFMODULE template="#request.ecm.AppPath#ecmShowRandomSummary.cfm"

               ID=""

>
```

This function is basically the same as the Random Summary function in ASP. See "Random Summary" on page 122 for details.

# Search Function

The search tag is used to display a text box, with a search button on the template. When seen on the template on the Web page, the users will be allowed to enter text and click on the search button to search through all the content on the Web site.

Shown here is the format for the ecmSearch tag.

```
   <CFMODULE template="#request.ecm.AppPath#ecmSearch.cfm"

               StartingFolder=""

               Recursive=""

               TargetPage=""

               TextBoxSize=""

               MaxCharacters=""

               ButtonImageSrc=""

               ButtonText=""

               FontFace=""

               FontColor=""
```

```
                FontSize=""

                Horizontal=""

        >
```

This function is basically the same as the Search function in ASP. See ”Search Function” on page 77 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

### Search Tag Example

```
<tr>

    <td>

        <CFMODULE template="#request.ecm.AppPath#ecmSearch.cfm"

            StartingFolder="\marketing" Recursive="1" TargetPage="search-

            display.cfm" TextBoxSize="25" MaxCharacters="200"

            ButtonImageSrc="" ButtonText="Search" FontFace="Verdana"

            FontColor="##808080" FontSize="2" Horizontal="0">

    </td>

</tr>
```

## Search Display Function

The search display custom tag created the area where all the search results show after the search has been executed.

Shown here is the format of the ecmSearchDisplay tag:

```
        <CFMODULE template="#request.ecm.AppPath#ecmSearchDisplay.cfm"

                MaxNumber=""

                StyleInfo=""

                ShowDate=""

        >
```

This function is basically the same as the Search Display function in ASP. See ”Search Display Function” on page 87 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

### Search Display Example

```
<tr>

    <td>

        <CFMODULE template="#request.ecm.AppPath#ecmSearchDisplay.cfm"

        MaxNumber="0" StyleInfo="" ShowDate="1">

    </td>

</tr>
```

## List Summary Function

The list summary function is used to display the list of content summaries on a Web page. Shown here is the format of the list summary function:

```
        <CFMODULE template="#request.ecm.AppPath#ecmListSummary.cfm"

                    Folder=""

                    MaxNumber=""

                    Recursive=""

                    ShowSummary=""

                    StyleInfo=""

                    OrderBy=""

                    ShowInfo=""

        >
```

This function is basically the same as the List Summary function in ASP. See ”List Summary Function” on page 93 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

### List Summary Example

```
<TR>

    <TD>

        <CFMODULE template="#request.ecm.AppPath#ecmListSummary.cfm"

        Folder="\Marketing\News" MaxNumber="0" Recursive="0"

        ShowSummary="1" StyleInfo="" OrderBy="Title,asc" ShowInfo="" >

    </TD>

</TR>
```

## XML List Summary Function

The XML List Summary function wraps XML around the properties of a list of content, created as a list summary in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the list summary information. The ecmListSummaryXML function allows you to easily define how the list summary data appears on the Web page.

```
<TR>

    <TD>

        <CFMODULE template="#request.ecm.AppPath#ecmListSummaryXML.cfm"

        Folder="" Recursive="" OrderBy=""

        MaxNumber="" ObjType="" SummaryType="" >

    </TD>

</TR>
```

This function is basically the same as the List Summary XML function in ASP. See ”XML List Summary Function” on page 312 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

### XML List Summary Example

```
<TR>

    <TD>

        <CFMODULE template="#request.ecm.AppPath#ecmListSummaryXML.cfm"

        Folder="\books\" Recursive="0" OrderBy="Title,asc"

         MaxNumber="0" ObjType="folder" SummaryType="alltypes">

    </TD>

</TR>
```

## RSS List Summary Function

The RSS summary function is used to create an XML list of content summaries for RSS Syndication. Shown here is the format of the RSS list summary function:

```
<CFMODULE template="#request.ecm.AppPath#ecmListSummary.cfm"

            Folder=""

            MaxNumber=""

            Recursive=""

            OrderBy=""

>
```

This function is basically the same as the RSS Summary function in ASP. See ”RSS Summary Function” on page 322 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

### List Summary Example

```
<TR>

    <TD>
```

```
        <CFMODULE template="#request.ecm.AppPath#ecmListSummary.cfm"

        Folder="\books" MaxNumber="0" Recursive="0"

        OrderBy="Title,asc" >

    </TD>

</TR>
```

# Archive List Summary Function

The Archive list summary function displays on a Web page a list of content summaries for archived content blocks. For example, you might have a current news page and an archived news page. Use the List Summary Function to display current news, and this function to display older news stories.

On the Content Block **schedule** screen, the user must set an **End Date** then choose the Archive and Display option to have the content blocks appear via this function.

This function has the same parameters as the List Summary Function. See ”List Summary Function” on page 384 for details.

# Single Summary Function

The single summary tag is used to display the summary of one content block in the Ektron CMS400.NET Web site, instead of a list of all the summaries in a content folder. Seen below is the single summary tag:

```
<cfmodule template="#request.ecm.AppPath#ecmSingleSummary.cfm"

            ContentID=""

            ShowSummary="1"

            StyleInfo=""

            ShowInfo=""

>
```

This function is basically the same as the Single Summary function in ASP. See ”Single Summary Function” on page 102 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

### Single Summary Example

```
<TR>

    <TD>

        <cfmodule template="#request.ecm.AppPath#ecmSingleSummary.cfm"

        ContentID="14" ShowSummary="1" StyleInfo=""

        ShowInfo="DateModified">

    </TD>

</TR>
```

# Collection Function

The collection function is used to display the list of links that were created as a collection in the Ektron CMS400.NET Workarea. The ecmCollection function is highly customizable, allowing you to easily define how the collection data will be displayed on the Web page.

Shown here is the format of the ecmCollection function.

```
<CFMODULE template="#request.ecm.AppPath#ecmCollection.cfm"

            id="2"

            displayFunction="ecmTeaser"

>
```

This function is basically the same as the Collection function in ASP. See ”Collection Function” on page 107 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

## Example Using the ecmNavigation Display Function

The following is an example of a collection function being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<CFMODULE id="1" displayFunction="ecmNavigation"

template="#request.ecm.AppPath#ecmCollection.cfm">
```

As stated earlier, the format of the collection on the Web page depends on the displayFunction that is used. Below is the source code for the display function “ecmNavigation”.

```
<cfparam name="Attributes.nId" default="">

<cfparam name="Attributes.href" default="">

<cfparam name="Attributes.target" default="">

<cfparam name="Attributes.startStyle" default="">

<cfparam name="Attributes.endStyle" default="">

<cfset LocalError = "">

<cfobject class="CMS400.NETContentBrv2.CMS400.NETContent" Action="Create" name="cmsObj">

<cfset cNavs =
cmsObj.GetEcmCollectionItems(request.ecm.AppConfStr,Attributes.nId,0,request.ecm.cmsPreview,r
equest.ecm.user_id,request.ecm.site_id,LocalError)>

<cfif cNavs.Count()>

    <cfset nString = "">

    <cfloop collection=#cNavs# Item=nav>

        <cfset nString = nString & "<a href="""">

        <cfif len(Attributes.href)>

            <cfif FindNoCase("?", Attributes.hrefs)>

                <cfset nString = nString & Attributes.href & "&id=" &

                nav.Item("ContentID")>

            <cfelse>

                <cfset nString = nString & Attributes.href & "?id=" &

                nav.Item("ContentID")>

            </cfif>
```

```
        <cfelse>

            <cfset nString = nString & nav.Item("ContentLinks")>

        </cfif>

        <cfset nString = nString & """ ">

        <cfif len(Attributes.target)>

            <cfset nString = nString & "target=""" & Attributes.target & """ ">

        </cfif>

        <cfset nString = nString & ">">

        <cfif len(Attributes.startStyle)>

            <cfset nString = nString & Attributes.startStyle>

        </cfif>

        <cfset nString = nString & nav.Item("ContentTitle")>

        <cfif len(Attributes.endStyle)>

            <cfset nString = nString & Attributes.endStyle>

        </cfif>

        <cfset nString = nString & "</a>">

    </CFLOOP>

        <cfmodule str="#nString#" nId="#Attributes.nId#"
template="#request.ecm.AppPath#ecmshowCollectionsContent.cfm">

</cfif>

<cfset nNavs = "Nothing">

<cfset cmsObj = "Nothing">
```

As you can see, the ecmNavigation function is a simple ASP function that creates a table of the links that belong to the collection specified. The function also displays the title of the content blocks by using the “info(“ContentTitle”)) building block.

## Example Using the ecmTeaser Display Function

The following is an example of a collection function being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<CFMODULE id="2" displayFunction="ecmTeaser"

template="#request.ecm.AppPath#ecmCollection.cfm">
```

In this example, the collection with an ID=2 is displayed, and the function “ecmTeaser” defines how to display the collection data on a Web page. Here is how the collection would appear on a Web page.

RC International
Home | Products | Support | News | Careers | Search Login | Contact Ektron
Employment Opportunities
Plastic Molder #123 (8/27/2002 2:38:34 PM)
RC International is looking for an experienced plastics molder.
Servo Control Engineer #124 (8/27/2002 2:40:26 PM)
RC International is looked for a highly skilled servo control engineer to join our team.

As stated earlier, the format of the collection on the Web page depends on the displayFunction that is used. Below is the source code for the display function “ecmTeaser”.

```
<cfparam name="Attributes.nID" default="">

<cfparam name="Attributes.href" default="">

<cfparam name="Attributes.ShowSummary" default="">

<cfparam name="Attributes.StyleInfo" default="">

<cfparam name="Attributes.target" default="">

<cfset ErrorString = "">

<cfobject class="CMS400.NETContentBrv2.CMS400.NETContent" Action="Create" name="tObj">

<cfset cTeasers = tObj.GetEcmCollectionItems(request.ecm.AppConfStr, Attributes.nId, 0,
request.ecm.cmsPreview, request.ecm.user_id, request.ecm.site_id, ErrorString)>

<cfset tString = "<table border=""0"" cellspacing=""0"" cellpadding=""0"" width=""100%"">">

<cfif cTeasers.Count()>
```

```
    <cfloop collection=#cTeasers# Item=cT>

        <cfset tString = tString & "<tr><td>">

        <cfset tString = tString & "<a href=""">

        <cfif len(Attributes.href)>

            <cfif FindNoCase("?", Attributes.hrefs)>

                <cfset tString = tString & Attributes.href & "&id=" &

                cT.Item("ContentID")>

            <cfelse>

                <cfset tStringi = tString & Attributes.href & "?id=" &

                cT.Item("ContentID")>

            </cfif>

        <cfelse>

            <cfset tString = tString & cT.Item("ContentLinks")>

        </cfif>

        <cfset tString = tString & """ ">

        <cfif len(Attributes.target)>

            <cfset tString = tString & "target=""" & Attributes.target & """ ">

        </cfif>

        <cfset tString = tString & ">">

        <cfset tString = tString & cT.Item("ContentTitle") & "</a>" & " (" &

        DateFormat(cT.Item("DateModified"), "mm/dd/yyyy") & " " &

        TimeFormat(cT.Item("DateModified"), "h:mm:ss tt") & ")</td></tr><tr><td>" &

        cT.Item("ContentTeaser")>

        <cfset tString = tString & "</td></tr><tr><td> </td></tr>">

    </cfloop>

    <cfset tString = tString & "</table>">

        <cfmodule str="#tString#" nId="#Attributes.nId#"
template="#request.ecm.AppPath#ecmshowCollectionsContent.cfm">

</cfif>
```

As you can see, the ecmTeaser function is a simple ASP function that creates a table of the links that belong to the collection specified.

The ecmTeaser function also displays the:

- Content Title
- Date Modified
- Content Teaser

### Collection Variables

See ”Collection Attributes” on page 112

### DisplayFunction

The display function defines how collection data appears on a Web page. Using ColdFusion scripting, you can create your own display functions, or use the three included functions.

#### *Building Blocks*

See ”Building Blocks” on page 116

## XML Collection Function

The XML collection function wraps XML around the parameters of a list of content that’s created as a collection in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the collection information. The ecmCollectionXML function allows you to easily define how the collection data appears on the Web page.

Shown here is the format of the ecmCollection function.

```
<CFMODULE template="#request.ecm.AppPath#ecmCollectionXML.cfm"

          id=""

          getHtml=""

>
```

This function is basically the same as the XML Collection function in ASP. See "XML Collection Function" on page 343 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

```
<CFMODULE template="#request.ecm.AppPath#ecmCollectionXML.cfm" id="3" getHtml="0">
```

# RSS Collection Function

The RSS collection function is used to create an XML list of links that were created as a collection in the Ektron CMS400.NET Workarea and display them in RSS Syndication.

Shown here is the format of the ecmRssCollection function.

```
<CFMODULE template="#request.ecm.AppPath#ecmRssCollection.cfm"

          id="2"

>
```

This function is basically the same as the RSS Collection function in ASP. See "RSS Collection Function" on page 341 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it.

# Event Calendar Function

The ecmEvtCalendar function is used to display an event calendar that has been created in the Ektron CMS400.NET Workarea.

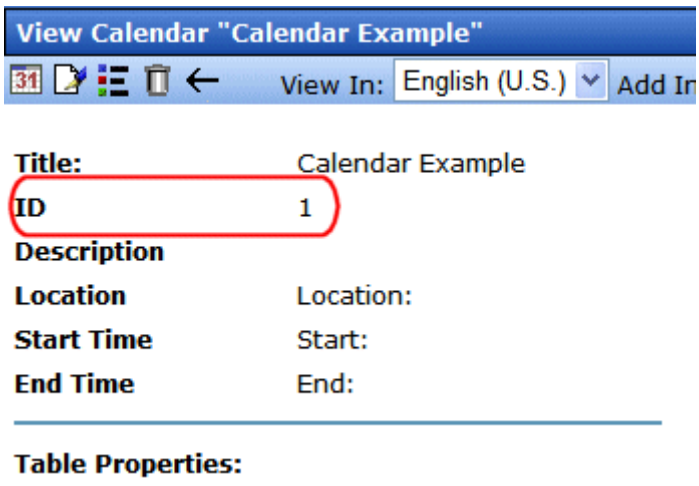
Displayed here is the format for the event calendar custom function:

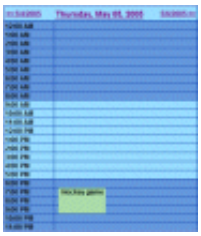
```
<CFMODULE template="#request.ecm.AppPath#ecmEvtCalendar.cfm" Calendar_ID="" DisplayType="">
```

This function is basically the same as the Event Calendar function in ASP. See "Event Calendar Function" on page 123 for details.

The only differences between the Coldfusion function and the ASP one is the use of the DisplayType parameter and the syntax for

implementing it. The ecmEvtCalendar function parameters are defined below.

| Attribute | Description |
|---|---|
| id | The **"id"** argument represents the ID number of the event calendar in the Workarea.<br>  |

| Attribute | Description |
|---|---|
| DisplayType | Specify the Calendar display type.<br>• **Month** - A 30 day calendar month view.<br> <br>• **Day** - Single day view.<br>  <br>• **Inline** - Display events that occur from the start date of the calendar through the end date.<br> <br>• **MonthUpWithEvent** - Displays a month calendar. Events appear to the right of the calendar when you click on a day.<br> |

Here is a sample of the Event Calendar syntax.

### Event Calendar Sample

```
    <tr>

         <td>

             <CFModule template="#request.ecm.AppPath#ecmEvtCalendar.cfm" calendar_id="24"
DisplayType="monthupwithevent">

         </td>

    </tr>
```

# Form Function

The ecmFormBlock function displays an HTML form content block. For more information about HTML forms, see the **Ektron CMS400.NET** Administrator manual chapter “Working with HTML Forms.”

**IMPORTANT!** If you create a template for an existing form content block, you must manually change its quicklink to point to the new template. This change does not occur automatically. This procedure is described in the Ektron CMS400.NET Administrator manual chapter “Managing Library Assets”, section “Updating Default Template for Multiple Quicklinks.”

Here is the format for the form custom function

```
<CFMODULE template="#request.ecm.AppPath#ecmFormBlock.cfm" ID="" >
```

This function is basically the same as the Form function in ASP. See ”Form Function” on page 129 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion example is below.

Here is an example of a form content block in an Ektron CMS400.NET template.

```
    <tr>

         <td>

                 <CFModule template="#request.ecm.AppPath#ecmFormBlock" form_id="1">
```

```
        </td>
    </tr>
```

# MultiPurpose Function

The ecmMultiPurpose function displays either a content block or an content block associated with a form. As a result, you can use one template to display both types of content block.

Here is the format for the Multipurpose function

```
<CFMODULE template="#request.ecm.AppPath#ecmMultiPurpose.cfm"  DefaultID="" >
```

To pass in a standard content block, pass the id value in the url parameter, as illustrated below.

```
http://localhost/CMS400Example/index.asp?id=2
```

To pass in a Form content block, pass the ekfrm value in the url parameter, as illustrated below.

```
http://localhost/CMS400Example/index.asp?ekfrm=2
```

**NOTE** For more information about passing the id value as a url parameter, see "Dynamic Content Block" on page 381.

If the id value or ekfrm value does not exist, a blank template appears.

If the url parameter is not passed in or recognized, the function calls ecmContentBlock and passes the DefaultID value as the id argument. If the DefaultID value is zero (0), ecmContentBlock is not called.

# Metadata Function

The metadata tag is used to display the metadata for one or more content blocks in the Ektron CMS400.NET Web site. Inserting this custom tag will allow you to generate the list of metadata that is edited by the authors, and insert it in the source of the Web page.

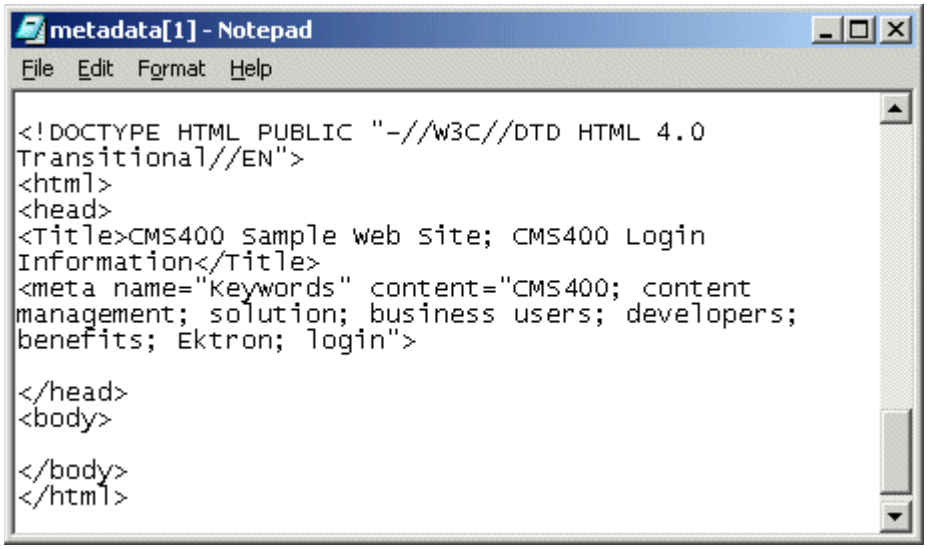
Shown below is an example of what the metadata custom tag looks like.

```
<cfmodule template="#request.ecm.AppPath#ecmMetadata.cfm" ItemList="" >
```

This function is basically the same as the Metadata function in ASP. See "Metadata Function" on page 132 for details.

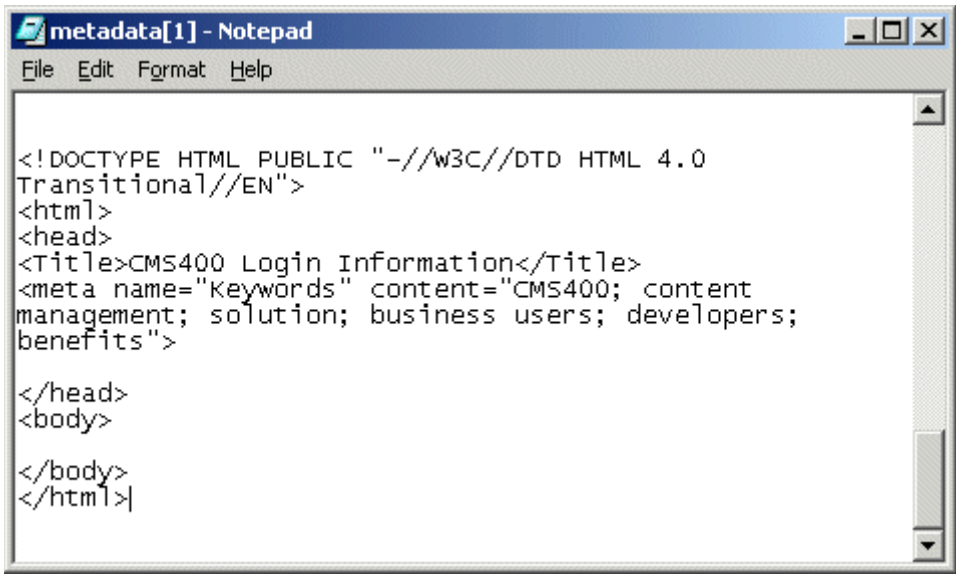
The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The ColdFusion examples are below.

```
<cfmodule template="#request.ecm.AppPath#ecmMetadata.cfm" ItemList="[1][2]">
```

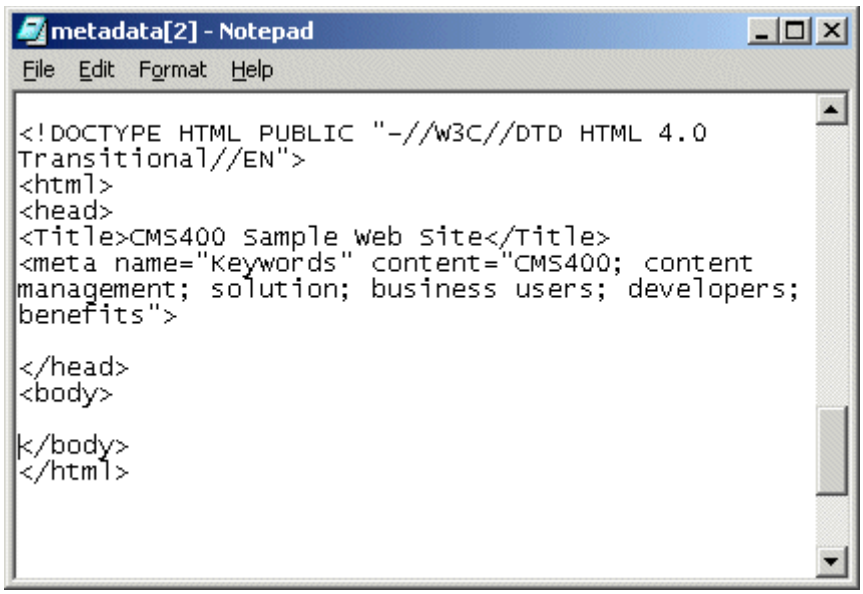



```
<cfmodule template="#request.ecm.AppPath#ecmMetadata.cfm"
```

```
ItemList="[1;title][2;keywords]>
```




```
<cfmodule template="#request.ecm.AppPath#ecmMetadata.cfm"
ItemList="[1][2;title,keywords]>
```



# DHTML Menus

**Ektron CMS400.NET**'s Menu feature lets users create and maintain a dropdown menu system for your Web site. The Menu feature lets you display menu options that link to content blocks, library assets, external hyperlinks and submenus.

To learn more about menus, please refer to the "Menu Feature" chapter of the **Ektron CMS400.NET** User Manual.

## Additional Documentation in Ektron Knowledge Base

This chapter explains how to implement standard menus that ship with **Ektron CMS400.NET**. Ektron's Web site also provides Knowledge Base articles that explain how to extend the standard menu features.

## Displaying a Menu on a Web Page

### *GenerateMenu and DropMenu*

**IMPORTANT!** The menu functions described below are only one example of how to implement menus. You can implement your own menu system using JavaScript.

Use the `GenerateNextGenMenu` and `DropNextGenMenu` custom functions to display a DHTML menu on a Web page. Place `GenerateNextGenMenu` once on the page to provide an infrastructure for `DropNextGenMenu`. Then, place `DropNextGenMenu` as needed to implement individual menus.

Follow these steps to display a menu on a Web template.

1. Place the following code directly after the page's `<body>` tag.

```
<CFMODULE Ids="menu id number"
template="#request.ecm.AppPath#ecmDHTML_GenerateNextGenMenu.cfm">
```

For example:

```
<CFMODULE Ids="1" template="#request.ecm.AppPath#ecmDHTML_GenerateNextGenMenu.cfm">
```

To display more than one menu, separate each id number with a comma. For example:

```
<CFMODULE Ids="1,2,3" template="#request.ecm.AppPath#ecmDHTML_GenerateNextGenMenu.cfm">
```

2. If the user clicks *text* to display the menu, place the following code within the template file where the text would appear.

```
<CFMODULE MenuId="menu id number" Title="text user clicks to display menu"
template="#request.ecm.AppPath#ecmDHTML_DropNextGenMenu.cfm">
```

For example:

```
<CFMODULE MenuId="1" Title="Products"
template="#request.ecm.AppPath#ecmDHTML_DropNextGenMenu.cfm">
```

If the user clicks *a graphic* to invoke the menu, use this syntax:

```
<CFMODULE MenuId="menu id number" Title= '<img src=graphic file name>'
template="#request.ecm.AppPath#ecmDHTML_DropNextGenMenu.cfm">
```

For example:

```
<CFMODULE MenuId="1" Title= '<img src=mymenugraphic.jpg>'
template="#request.ecm.AppPath#ecmDHTML_DropNextGenMenu.cfm">
```

### Customizing the Menu's Appearance

You can customize a menu's appearance, such as the font style, color, and size. So, you have the flexibility to coordinate a menu with the design your Web site.

To customize a menu in **Ektron CMS400.NET**. You can change:

- ek_Menu.js

### ek_Menu.js

You can change the look and feel of a menu by changing the ek_menu.js file. If you used the default settings when installing **Ektron CMS400.NET**, this file is in *webroot*`/CMS400Example/Workarea/java/`. For a description of the ek_menu.js file, see "ek_Menu.js" on page 372.

## XML Menu Function

The ecmGetMenuXML function wraps XML around the properties of a menu created in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the menu. The

ecmGetMenuXML function allows you to easily define how the menu data appears on the Web page.

```
<CFMODULE id="" template="#request.ecm.AppPath#ecmGetMenuXML.cfm">
```

This function is basically the same as the XML Menu function in ASP. See "XML Menu" on page 375 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it. The Coldfusion example is below.

```
<CFMODULE id="1" template="#request.ecm.AppPath#ecmGetMenuXML.cfm">
```

# Language Select Function

This function lets a visitor to your Web site select a language in which to view the site. Here is what the function looks like when published on a Web page.

Shown here is the format of the ecmLanguageSelect function.

```
<CFMODULE template="#request.ecm.AppPath#ecmLanguageSelect.cfm">
```

This function is basically the same as the Show Site Language function in ASP. See "Show Site Language Function" on page 369 for details.

The only difference between the Coldfusion function and the ASP one is the syntax for implementing it.

# Custom PHP Functions

Ektron CMS400.NET maintains and displays content via custom Ektron PHP functions and ComObjects. For the custom PHP functions to work properly, do the following:

- Make sure the `php_soap.dll` file is in the `c:\php` folder on the server where you installed PHP.
- Add the following line to the php.ini file.

  ```
  extension=php_soap.dll
  ```

  The php.ini file is located in `C:\Windows\php.ini`.
- In the `ApplicationAPI.php` file, set the WSDLpath. The `applicationAPI.php` file is located by default in:

  ```
  http://localhost/CMS400Example/WorkArea/platforms/
  PHP/applicationAPI.php
  ```

**IMPORTANT!** `ContentWS.asmx` is the web services (WSDL) file that the custom PHP functions use to talk to **Ektron CMS400.NET**. The WSDLpath located in the `applicationAPI.php` file must be set to location of the WSDL file. The default WSDLpath is `http://localhost/CMS400Example/WorkArea/WebServices/ContentWS.asmx`. If you have changed the location of `ContentWS.asmx` you need to update the WSDLpath. For example, if your site is `www.myektronsite.com`, the WSDLpath would be `http://www.myektronsite.com/WorkArea/WebServices/ContentWS.asmx`.

- Add the following line to the top of each template.

  ```
  <?php include 'applicationAPI.php'; ?>
  ```

  This file calls the Ektron custom functions file to allow the page to function properly. It *must* appear at the top of the template file.

To customize you Ektron CMS400.NET application, you need to be familiar with these functions.

| Function | Description | More Information |
|---|---|---|
| **Calendar Functions** | | |

| Function | Description | More Information |
|---|---|---|
| Event Calendar | Displays a month view event calendar | "Event Calendar Function" on page 419 |
| **Collection Functions** | | |
| Collection | Displays links generated by collections | "Collection Function" on page 413 |
| CollectionXML | Creates XML for a collection by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed. | "XML Collection Function" on page 417 |
| Random Content | Randomly displays the content of a collection item | "Random Content Function" on page 418 |
| Random Summary | Randomly displays the summary of a collection item | "Random Summary Function" on page 419 |
| RSS Collection | Creates an RSS feed out of the items in a collection. | "RSS Collection Function" on page 418 |
| **Content Functions** | | |
| Content Block | Displays and manages a content block | "Content Block Function" on page 407 |
| Form | Displays a content block associated with a form | "Form Function" on page 422 |
| Multipurpose | Displays either a content block or a form | "MultiPurpose Function" on page 423 |
| **List Summary Functions** | | |
| Archive ListSummary | Displays summaries of archived content blocks | "Archive List Summary Function" on page 411 |
| List Summary | Displays a list of content block summaries | "List Summary Function" on page 410 |

| Function | Description | More Information |
|---|---|---|
| List Summary XML | Creates XML for a list summary by wrapping tags around all of the collection items and their properties. This allows you to use an XSLT to define how the XML collection is displayed. | "XML List Summary Function" on page 412 |
| RSS List Summary | Creates an RSS feed out of the items in a list summary. | "RSS List Summary Function" on page 413 |
| Single Summary | Displays the summary for a content block | "Single Summary Function" on page 411 |
| **Login Functions** | | |
| Login | Adds the login or logout button | "Login Function" on page 407 |
| **Menu Functions** | | |
| DropHTMLMenu | Used to display a DHTML menu | "DHTML Menus" on page 426 |
| GenerateMenu | Works with DropHTML menu to display DHTML menus | "DHTML Menus" on page 426 |
| XML Menu | Creates XML for a menu by wrapping tags around all of the menu items properties. | "XML Menu Function" on page 428 |
| **Metadata Functions** | | |
| Metadata | Displays the metadata for the current Web page | "Metadata Function" on page 423 |
| **Search Functions** | | |
| Search | Inserts a text box with a search button | "Search Function" on page 408 |
| Search Display | Creates an area to display search results | "Search Display Function" on page 409 |
| **Language Functions** | | |
| Language Select | Lets a site visitor select a language in which to view site. | "Language Select Function" on page 428 |

# Login Function

This function is responsible for displaying the following buttons on the Web page.

This function is basically the same as the Login function in ASP. See "Login Function" on page 46 for details. The only difference is the syntax used to implement the function. This is listed below.

### Login Function Example

```
<tr>

    <td>

        <?php ecmlogin(); ?>

    </td>

</tr>
```

# Content Block Function

The content block and content blockEX functions are used to display a content block on an Ektron CMS400.NET Web page.

Shown here is the format for the content block function:

```
<?php ecmContentBlock (ID) ?>
```

Here is the format for the content blockEX function:

```
<?php ecmContentBlockEx (ID, "XSLT", Override) ?>
```

**NOTE** As an alternative, you can use the multipurpose function, which can display either a content block or a content block associated with a form. For more information, see "MultiPurpose Function" on page 423.

This function is basically the same as the Add Content function in ASP. See "Content Block Function" on page 49 for details.

The only differences between the PHP function and the ASP one are the examples of implementing it. The PHP examples are below.

### Static Content Block

```
<tr>

   <td>

      <?php  ecmContentBlock(1) ?>

   </td>

</tr>
```

### Dynamic Content Block

```
<tr>

   <td>

      <?php  ecmContentBlock($id) ?>

   </td>

</tr>
```

### XML Content Block

```
<tr>

    <td>

       <?php ecmContentBlockEx (13, “”, 1) ?>

    </td>

</tr>
```

## Search Function

The search function is used to display a text box with a search button on the template. When seen on the template on the Web page, a user can enter text and click the search button to search through all the content on the Web site.

Here is the format for the ecmSearch function.

```
<?php ecmSearch( “StartingFolder”,

       Recursive,

       "targetpage",

       textbox,
```

```
        MaxCharacters,

        "ButtonImgSrc",

        "ButtonText",

        "FontFace",

        "FontColor",

        "FontSize",

        Horizontal,

        "Spare" );

?>
```

This function is basically the same as the Search function in ASP. See ”Search Function” on page 77 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

### Search Function Example

```
<TR>

 <TD>

  <?php ecmSearch("\\Marketing",1,"search-
  display.php",25,200,"","Search","Verdana","#808080","2",0,""); ?>

 </TD>

</TR>
```

# Search Display Function

The search display custom function created the area where all the search results show after the search has been executed.

Here is the format of the ecmSearchDisplay function:

```
<?php ecmSearchDisplay( MaxNumbers,"StyleInfo", ShowDate ); ?>
```

This function is basically the same as the Search Display function in ASP. See ”Search Display Function” on page 87 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

### Search Display Function Example

```
<tr>

   <td>

      <?php ecmSearchDisplay( 0,"",1); ?>

   </td>

</tr>
```

## List Summary Function

The list summary function is used to display the list of content summaries on a Web page. Here is the format of the list summary function.

```
<?php ecmListSummary ( "Folder",

       Recursive,

       ShowSummary,

       "StyleInfo",

       "OrderBy",

       "ShowInfo",

       MaxNumber,

       "Spare 1",

       "Spare2");

?>
```

This function is basically the same as the List Summary function in ASP. See ”List Summary Function” on page 93 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

### List Summary Example

```
<TR>
```

```
    <TD>

        <?PHP ecmListSummary ("\Marketing\News",0,1,"",

        "Title,asc","DateModified",0,"","",""); ?>

    </TD>

</TR>
```

# Archive List Summary Function

The Archive list summary function displays on a Web page a list of content summaries for archived content blocks. For example, you might have a current news page and an archived news page. Use the List Summary Function to display current news, and this function to display older news stories.

On the Content Block **schedule** screen, the user must set an **End Date** then choose the Archive and Display option to have the content blocks appear via this function.

This function has the same parameters as the List Summary Function. See ”List Summary Function” on page 410 for details.

# Single Summary Function

The single summary function is used to display the summary of one content block in the Ektron CMS400.NET Web site, instead of a list of all the summaries in a content folder. Seen below is the single summary function:

```
<?php ecmSingleSummary( “ContentID”,

        ShowSummary,

        “StyleInfo”,

        “ShowInfo”,

        “Spare1”);

?>
```

This function is basically the same as the Single Summary function in ASP. See ”Single Summary Function” on page 102 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

### Single Summary Example

```
<TR>

   <TD>

       <?PHP ecmSingleSummary ("14",1,"","DateModified",""); ?>

   </TD>

</TR>
```

# XML List Summary Function

The XML List Summary function wraps XML around the properties of a list of content, created as a list summary in the Ektron CMS400.NET Workarea. Once created you can use the to display the list summary information. The ecmListSummaryXML function allows you to easily define how the list summary data appears on the Web page.

**<?php ecmListSummaryXML (“Folder”, Recursive, “OrderBy”, MaxNumber, “ObjType”, “SummaryType”); ?>**

This function is basically the same as the List Summary XML function in ASP. See ”XML List Summary Function” on page 312 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

```
<?php ecmListSummaryXML("\books\\",0,"Title,asc",0,"folder",""); ?>
```

# RSS List Summary Function

The RSS summary function is used to create an XML list of content summaries for RSS Syndication. Shown here is the format of the RSS list summary function:

```
<?php ecmRssSummary (“Folder”, Recursive, “OrderBy”, MaxNumber); ?>
```

This function is basically the same as the RSS Summary function in ASP. See ”RSS Summary Function” on page 322 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

```
<?php  ecmRssSummary("\Books\\", 0, "Title,asc", 0); ?>
```

# Collection Function

The collection function is used to display the list of links that were created as a collection in the Ektron CMS400.NET Workarea. The ecmCollection function is highly customizable, allowing you to easily define how the collection data will be displayed on the Web page.

Shown here is the format of the ecmCollection function.

```
<?PHP ecmCollection( “id”,

    “displayFunction”;

?>
```

This function is basically the same as the Collection function in ASP. See ”Collection Function” on page 107 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

## Collection Examples

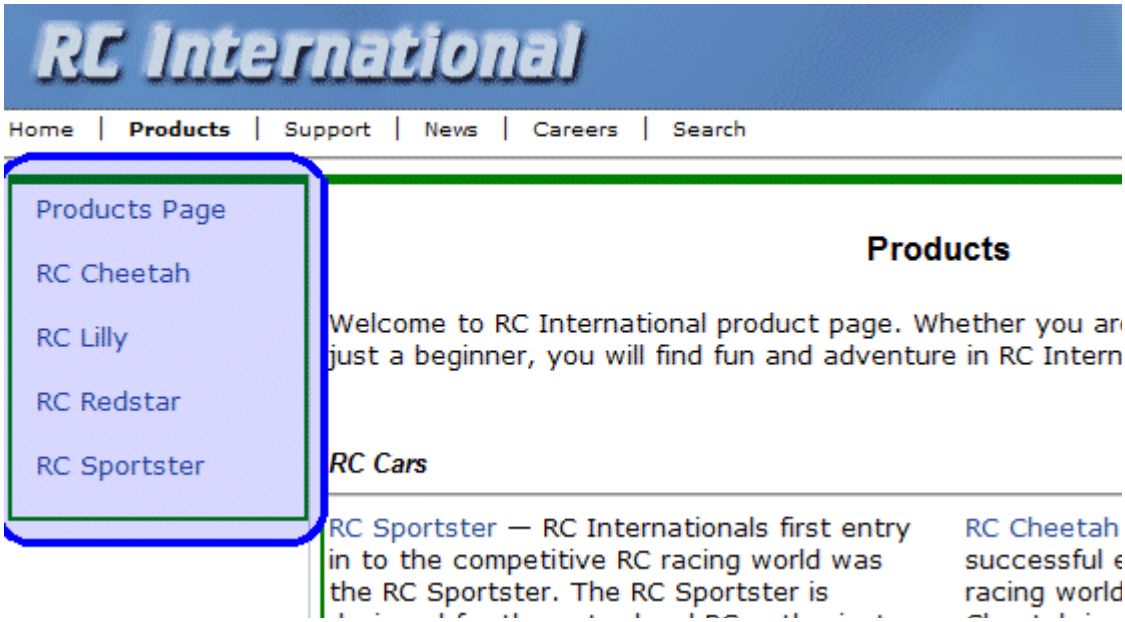
### *Example Using the ecmNavigation Display Function*

The following is an example of a collection function being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<?php ecmCollection(1, "ecmNavigation") ?>
```

In this example, the collection with an ID=1 is displayed, and the function “ecmNavigation” is used to define how the collection data is displayed on the Web page.

Here is how the collection would appear on the Web page.


RC International
Home | Products | Support | News | Careers | Search
Products Page
RC Cheetah
RC Lilly
RC Redstar
RC Sportster
Products
Welcome to RC International product page. Whether you ar
just a beginner, you will find fun and adventure in RC Intern
RC Cars
RC Sportster — RC Internationals first entry in to the competitive RC racing world was the RC Sportster. The RC Sportster is
RC Cheetah
successful e
racing world

The format of the collection on the Web page depends on the displayFunction that is used.

Below is the source code for the display function “ecmNavigation”.

```
Function ecmNavigation($cInfo){

    global $html, $info;

    $html = "<table border=\"0\" cellspacing=\"0\" cellpadding=\"0\"

        width=\"100%\">";

    $mycount = $cInfo->count();

    for ($iloop=0; $iloop < $mycount; ++$iloop) {

        $info = $cInfo->Item($iloop+1);

        $html = $html."<tr><td>  <a href=\"";

        $html = $html.$info->Item("ContentLinks");

        $html = $html."\">";

        $html = $html.$info->Item("ContentTitle");

        $html = $html."</a></td></tr><tr><td> </td></tr>";

    }

    $html = $html."</table>";

    return $html;

}
```

As you can see, the ecmNavigation function is a simple PHP function that creates a table of the links that belong to the collection specified.

The ecmNavigation function also displays the title of the content blocks by using the “info(“ContentTitle”)) building block.

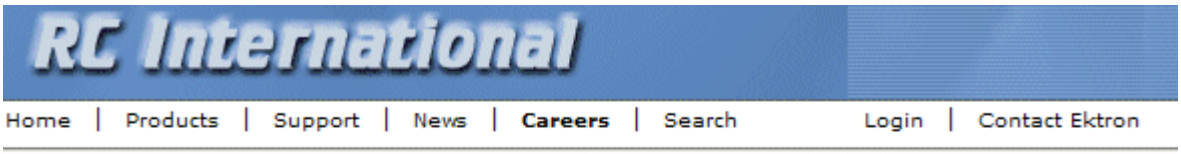
### *Example Using the ecmTeaser Display Function*

The following is an example of a collection function being used as a navigation menu in the Ektron CMS400.NET sample Web site.

```
<?php ecmCollection (2, "MyDisplayFunction"); ?>
```

In this example, the collection with an ID=2 is displayed, and the function “ecmTeaser” is used to define how the collection data is displayed on the Web page.

Shown here is how the collection would appear on the Web page.




**Employment Opportunities**

Plastic Molder #123 (8/27/2002 2:38:34 PM)
RC International is looking for an experienced plastics molder.

Servo Control Engineer #124 (8/27/2002 2:40:26 PM)
RC International is looked for a highly skilled servo control engineer to join our team.

As stated earlier, the format of the collection on the Web page depends on the displayFunction that is used.

Shown below is the source code for the display function "ecmTeaser"

```
Function ecmTeaser($cInfo){

    global $html, $info;

    $html = "<table border=\"0\" cellspacing=\"0\" cellpadding=\"0\" width=\"100%\">";

    $mycount = $cInfo->count();

    for ($iloop=0; $iloop < $mycount; ++$iloop) {

        $info = $cInfo->Item($iloop+1);

        $html = $html."<tr><td><a href=\"";

        $html = $html.$info->Item("ContentLinks");

        $html = $html."\" >";

        $html = $html.$info->Item("ContentTitle");

        $html = $html."</a> (";

        $html = $html.date("m/d/Y g:i:s A", $info->Item("DateModified"));

        $html = $html.")</td></tr><tr><td>";

        $html = $html.$info->Item("ContentTeaser");

        $html = $html."</td></tr><tr><td> </td></tr>";
```

```
    }

    $html = $html."</table>";

    return $html;

}
```

As you can see, the ecmTeaser function is a simple ASP function that creates a table of the links that belong to the collection specified.

The ecmTeaser function also displays the:

- Content Title
- Date Modified
- Content Teaser

### Collection Variables

See ”Collection Attributes” on page 112

### DisplayFunction

As explained earlier, the display function is a function that you create that will define how the collection data will be displayed on the Web page.

By using simple PHP scripting, you can create your own display functions, or use the three included functions.

#### *Building Blocks*

See ”Building Blocks” on page 116

## XML Collection Function

The XML collection function wraps XML around the parameters of a list of content that’s created as a collection in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the collection information. The ecmCollectionXML function allows you to easily define how the collection data appears on the Web page.

Shown here is the format of the ecmCollection function.

```
<?php ecmCollectionXML (id, GetHtml); ?>
```

This function is basically the same as the XML Collection function in ASP. See "XML Collection Function" on page 343 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

```
<?php ecmCollectionXML (3,0); ?>
```

# RSS Collection Function

The RSS collection function is used to create an XML list of links that were created as a collection in the Ektron CMS400.NET Workarea and display them in RSS Syndication.

Shown here is the PHP format of the ecmRssCollection function.

```
<?php ecmRssCollection (id); ?>
```

This function is basically the same as the RSS Collection function in ASP. See "RSS Collection Function" on page 341 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it.

# Random Content Function

The random content function is used to randomly display the content of a content block that belongs in a specified collection.

Displayed here is the format for the random content function

```
<?php ShowRandomContent (id); ?>
```

This function is basically the same as the Random Content function in ASP. See "Random Content" on page 121 for details.

# Random Summary Function

SImilar to the random content function, the random summary function is used to randomly display the summary of a content block that belongs in a specified collection.

Displayed here is the format for the random content function

```
<?php ShowRandomSummary (id); ?>
```

This function is basically the same as the Random Summary function in ASP. See "Random Summary" on page 122 for details.

# Event Calendar Function

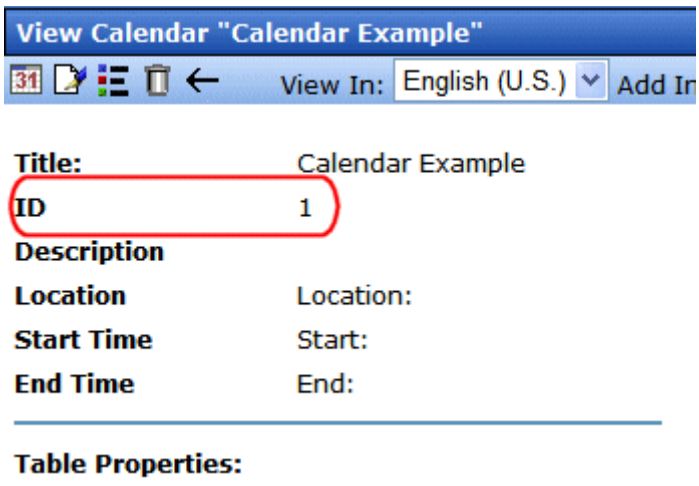
The ecmEvtCalendar function is used to display an event calendar that has been created in the Ektron CMS400.NET Workarea.

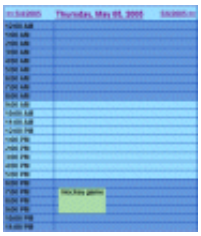
```
<?php ecmEvtCalendar(id, "Displaytype"); ?>
```

This function is basically the same as the Event Calendar function in ASP. See "Event Calendar Function" on page 123 for details.

The only differences between the PHP function and the ASP one is the use of the DisplayType parameter and the syntax for

implementing it. The ecmEvtCalendar function parameters are defined below.

| Attribute | Description |
|---|---|
| id | The **"id"** argument represents the ID number of the event calendar in the Workarea.<br>  |

| Attribute | Description |
|---|---|
| DisplayType | Specify the Calendar display type.<br>• **Month** - A 30 day calendar month view.<br> <br>• **Day** - Single day view.<br>  <br>• **Inline** - Display events that occur from the start date of the calendar through the end date.<br> <br>• **MonthUpWithEvent** - Displays a month calendar. Events appear to the right of the calendar when you click on a day.<br> |

### Event Calendar Sample

Here is an example of the event Calendar in an Ektron CMS400.NET template.

```
<tr>
    <td>
        <?php ecmEvtCalendar(24, “MonthUpWithEvent”); ?>
    </td>
</tr>
```

# Form Function

The ecmFormBlock function displays an HTML form content block. For more information about HTML forms, see the**Ektron CMS400.NET** Administrator manual chapter “Working with HTML Forms.”

**IMPORTANT!** If you create a template for an existing form content block, you must manually change its quicklink to point to the new template. This change does not occur automatically. This procedure is described in the Ektron CMS400.NET Administrator manual chapter “Managing Library Assets”, section “Updating Default Template for Multiple Quicklinks.”

Here is the format for the form custom function

**<?php ecmFormBlock(id); ?>**

This function is basically the same as the Form function in ASP. See ”Form Function” on page 129 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

```
<tr>
      <td>
           <?php ecmFormBlock(1); ?>
```

```
        </td>
    </tr>
```

**NOTE** As an alternative, you can use the multipurpose function, which can display either a content block or a content block associated with a form. For more information, see "MultiPurpose Function" on page 423.

# MultiPurpose Function

The ecmMultiPurpose function displays either a content block or a content block associated with a form. As a result, you can use one template to display both types of content block.

Here is the format for the Multipurpose function

**<?php ecmMultiPurpose(ID); ?>**

To pass in a standard content block, pass the id value in the url parameter, as illustrated below.

```
http://localhost/CMS400Example/index.asp?id=2
```

To pass in a Form content block, pass the ekfrm value in the url parameter, as illustrated below.

```
http://localhost/CMS400Example/index.asp?ekfrm=2
```

**NOTE** For more information about passing the id value as a url parameter, see "Dynamic Content Block" on page 408.

If the id value or ekfrm value does not exist, a blank template appears.

If the url parameter is not passed in or recognized, the function calls ecmContentBlock and passes the DefaultID value as the id argument. If the DefaultID value is zero (0), ecmContentBlock is not called.

# Metadata Function

The metadata function is used to display the metadata for one or more content blocks in the Ektron CMS400.NET Web site.

Inserting this custom function allows you to generate the list of metadata that is edited by the authors, and insert it in the source of the Web page. Shown below is an example of what the metadata custom function looks like.:

```
<?php ecmMetaData ("ItemList","Spare1";) ?>
```

This function is basically the same as the Metadata function in ASP. See ”Metadata Function” on page 132 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP examples are below.

To include both content blocks metadata in the web page, a ecmMetadata call would look like this.

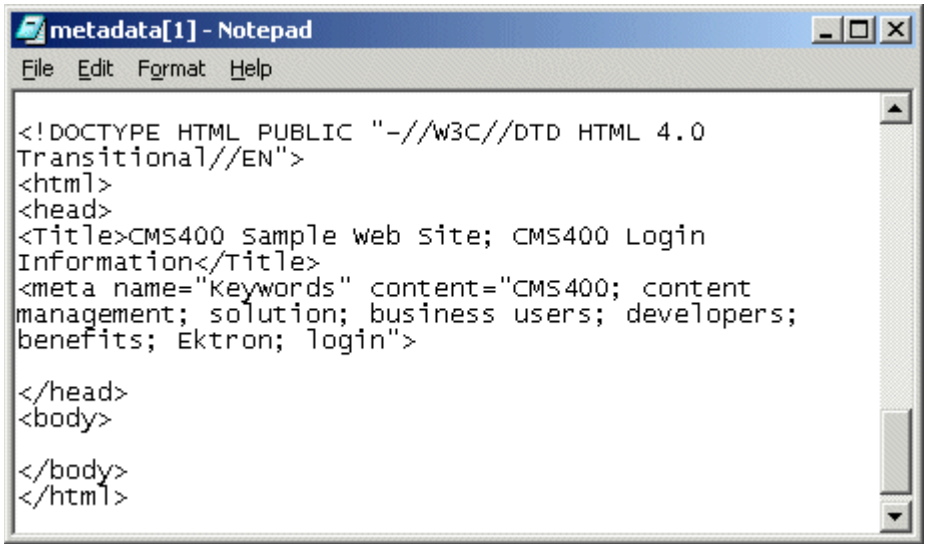
```
<html>
    <head>
        <?php ecmMetadata (“[1][2;Title,Content-Type]”, “”); ?>
        (rest of head)
    </head>
    <body>
        (rest of web page document)
    </body>
</html>
```

In the above example the ecmMetadata call is contained in the <head> of the HTML document (since all HTML metadata is required to exist there). Once the Web page is viewed the Web page source will contain the following information:

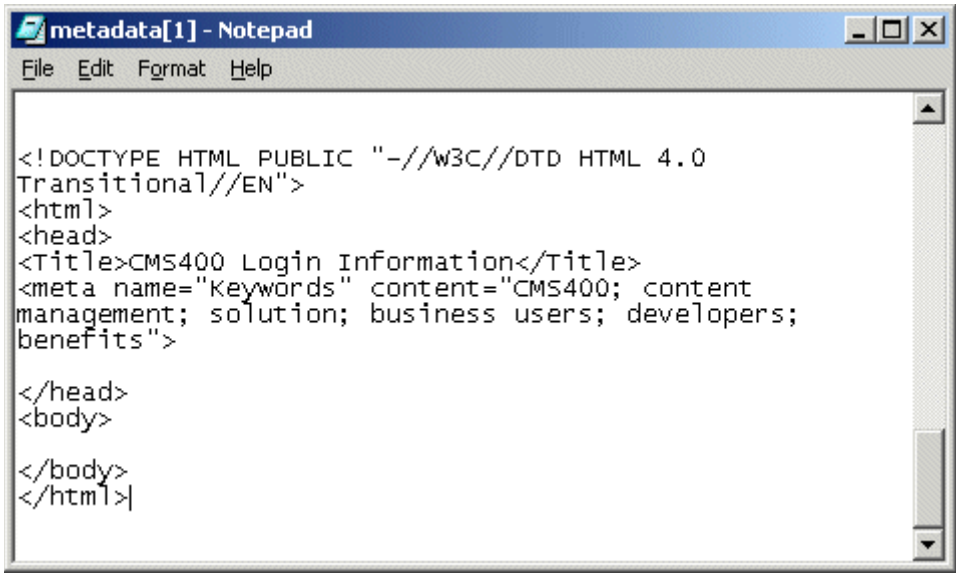
```
<html>
    <head>
        <Title>Welcome to Our Home Page</Title>
        <meta name=”Keywords” content= Home; page; company; Welcome; Announce;
        product; widget”>
        <meta http-equiv=”Content-Type” content=”text/html; charset=iso-8859-1”>
        (rest of head)
```

```
    </head>

    <body>

        (rest of web page document)

    </body>

</html>
```

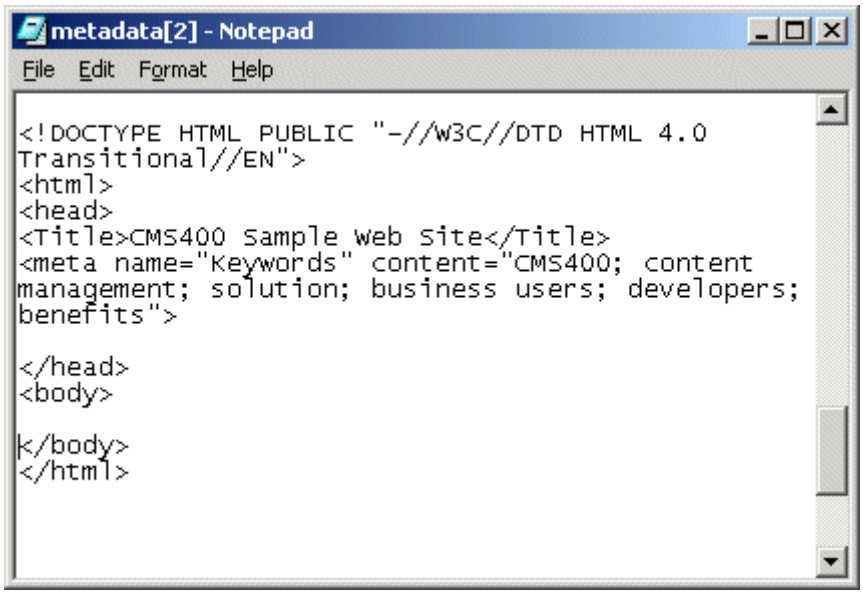
```
<?PHP ecmMetaData ("[1][2]","") ?>
```




```
<?PHP ecmMetaData ("[1;title][2;keywords]","") ?>
```



```
<?PHP ecmMetaData ("[1][2;title,keywords]","") ?>
```




# DHTML Menus

**Ektron CMS400.NET**'s Menu feature lets users create and maintain a dropdown menu system for your Web site. The Menu feature lets you display menu options that link to content blocks, library assets, external hyperlinks and submenus.

To learn more about menus, please refer to the "Menu Feature" chapter of the **Ektron CMS400.NET** User Manual.

## Additional Documentation in Ektron Knowledge Base

This chapter explains how to implement standard menus that ship with **Ektron CMS400.NET**. Ektron's Web site also provides Knowledge Base articles that explain how to extend the standard menu features.

## Displaying a Menu on a Web Page

### *GenerateMenu and DropMenu*

**IMPORTANT!** The menu functions described below are only one example of how to implement menus. You can implement your own menu system using JavaScript.

Use the `GenerateNextGenMenu` and `DropNextGenMenu` custom functions to display a DHTML menu on a Web page. Place `GenerateNextGenMenu` once on the page to provide an infrastructure for `DropNextGenMenu`. Then, place `DropNextGenMenu` as needed to implement individual menus.

Follow these steps to display a menu on a Web template.

1. Place the following code directly after the page's `<body>` tag.

```
<?php ecmDHTML_GenerateNextGenMenu("menu id number"); ?>
```

   For example:

```
<?php ecmDHTML_GenerateNextGenMenu("1"); ?>
```

   To display more than one menu, separate each id number with a comma. For example:

```
<?php ecmDHTML_GenerateNextGenMenu("1,2,3"); ?>
```

2. If the user clicks *text* to display the menu, place the following code within the template file where the text would appear.

```
<?php ecmDHTML_DropNextGenMenu menu ID number,"text user clicks to display menu"; ?>"
```

   For example:

```
<% ecmDHTML_DropNextGenMenu 1,"MyMenu1"; ?>
```

   If the user clicks *a graphic* to invoke the menu, use this syntax:

```
<?php ecmDHTML_DropNextGenMenu menu ID number, '<img src=graphic file name>'; ?>
```

   For example:

```
<?php ecmDHTML_DropNextGenMenu 1,'<img src=mymenugraphic.jpg>'; ?>
```

## Customizing the Menu's Appearance

You can customize a menu's appearance, such as the font style, color, and size. So, you have the flexibility to coordinate a menu with the design your Web site.

To customize a menu in **Ektron CMS400.NET**. You can change:

- ek_Menu.js

## ek_Menu.js

You can change the look and feel of a menu by changing the ek_menu.js file. If you used the default settings when installing

**Ektron CMS400.NET**, this file is in *webroot*/CMS400Example/Workarea/java/. For a description of the ek_menu.js file, see "ek_Menu.js" on page 372.

# XML Menu Function

The ecmGetMenuXML function wraps XML around the properties of a menu created in the Ektron CMS400.NET Workarea. Once created you can use the XML to display the menu. The ecmGetMenuXML function allows you to easily define how the menu data appears on the Web page.

```
<?php ecmGetMenuXML($Id); ?>
```

This function is basically the same as the XML Menu function in ASP. See "XML Menu" on page 375 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it. The PHP example is below.

```
<?php ecmGetMenuXML(1); ?>
```

# Language Select Function

This function lets a visitor to your Web site select a language in which to view the site. Here is what the function looks like when published on a Web page.

Shown here is the format of the ecmLanguageSelect function.

```
<?php ecmLanguageSelect(); ?>
```

This function is basically the same as the Show Site Language function in ASP. See ”Show Site Language Function” on page 369 for details.

The only difference between the PHP function and the ASP one is the syntax for implementing it.

# Custom API

As you have seen, the formatting for the custom functions used in Ektron CMS400.NET for PHP are defined in the file ApplicationAPI.php. As a developer, you may make changes to this file to further enhance how the custom functions appear on your Web site.

However, Ektron recommends creating a copy of the file, and renaming it to an arbitrary name, for example, MyCustomAPI.php, and including that file when creating Ektron CMS400.NET templates. This ensures that your custom API will not be overwritten if you upgrade to another version of Ektron CMS400.NET.

# Web Services

## Introduction to Web Services

Web Services (WS) are reusable software components that can be consumed by variety of clients, independent of operating system, programming language, or Web server, using standard Internet technologies. Because Web Services allow heterogeneous systems to communicate with each other in a standard way, they have received quick acceptance for data interchange over the Web.

It's easy for anyone to integrate your WS into their application. The client application can be a Web application, a Windows application, a Java application, a Visual Studio.net application, or a host of others -- it does not matter as long as it supports Web Services.

A Web Service consists of one or more methods that expose functionality. In this sense, a Web Service is an API that can be accessed using SOAP over HTTP. (SOAP is an XML-based, lightweight protocol that defines the message format of Web Services.)

For example, you might develop a purchasing application that automatically obtains price information from several vendors, lets the user select a vendor, submits the order and then tracks the shipment until it is received. The vendor application, in addition to exposing its services on the Web, might in turn use XML Web services to check a customer's credit, charge the customer's account and set up the shipment with a shipping company.

This manual does not explain how Web Services work. To learn more about them, try these sites.

- http://msdn.microsoft.com/webservices/understanding/readme/default.aspx

- http://aspalliance.com/jnuckolls/articles/introws/default.aspx
- http://aspnet.4guysfromrolla.com/articles/062602-1.aspx
- http://www.15seconds.com/Issue/010430.htm

### CMS400.NET's Implementation of Web Services

CMS400.NET supports XML Web Services. This means that you can create pages that offer your content to any Web-enabled device, and you can create pages that consume existing Web Services. CMS400.NET's Web Services feature can return data separated from presentation. In combination with XML support, this feature lets consumers retrieve your XML data independent of presentation information and act on it however they see fit.

### Benefits of Using Web Services

Web Services make it easy to connect with partners. They can also deliver more personal, integrated experiences to users via smart devices, including PCs. WSs can save time and money by reducing development time, and increase revenue by making your own XML Web services available to others.

## WS Components and CMS400.NET Architecture

Web Services require the installation of the .NET Framework on the server.

Ektron's Web Service, ContentWS.asmx, runs within the .NET Framework. Through the Application API class in the Ektron.cms.commonUI.dll, .NET managed code can communicate with asp.net assembly code. ContentWS.asmx utilizes the ektron.com.commonui.dll file to provide access to the Content Business Objects (content blocks, menus, collections, list summaries, etc.).

## Software and Hardware Requirements

- .NET framework, version 1.1 SP2, must be installed on the server (for more information, see http://msdn.microsoft.com/netframework/).

- Ektron CMS400.NET, version 4.5 or higher

# Installation of Files

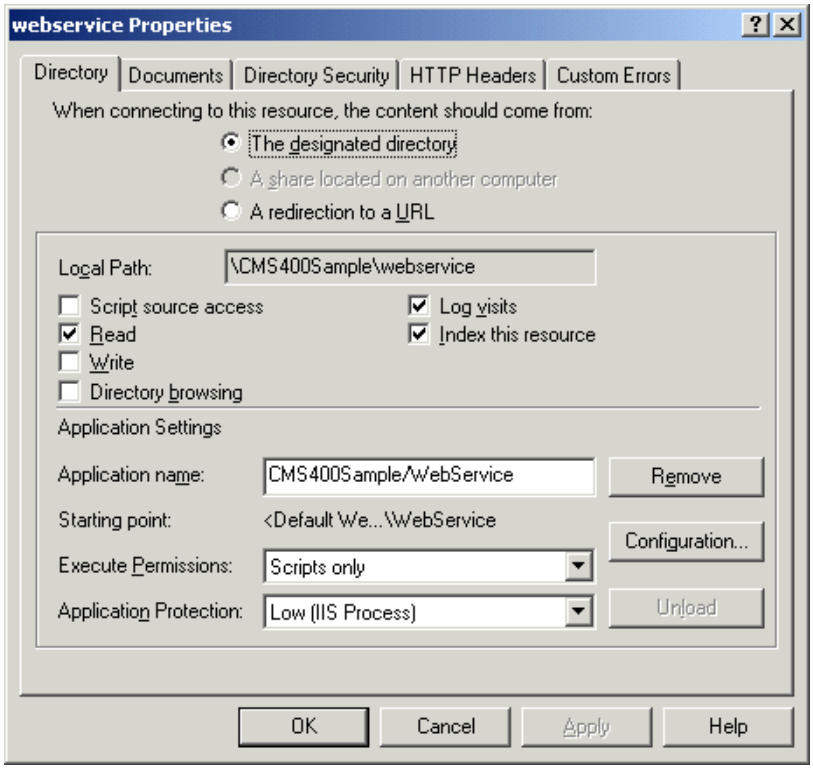
Below is a list of the files and their locations that make up the Web Services feature. They are installed when you install CMS400.NET.

| File or folder location | Description |
|---|---|
| *webroot*/CMS400Example/Workarea/webservices/ContentWS.asmx | Entry point for Web Service. This Web page displays examples of all methods. |
| *webroot*/CMS400Example/bin | Directory for compiled Web Service code |
| *webroot*/CMS400Example/bin/WebService.dll | Web Service dll |

# Setup Instructions

After you install CMS400.NET, you need to open IIS, select `CMS400Example/workarea`, and make it an application. To make a folder an application, follow these steps.

1. Open IIS.
2. Right click one of the folders listed above.
3. Select **Properties**. The Properties dialog appears.




4. Make sure you are on the **Directory** tab, which is selected by default.
5. Press the **Create** button next to the Application Name field.
6. Assign the folder's name in the **Application name** field.
7. Press **OK**.

If you are creating a new directory for your Web site, you must copy the following files into it from *webroot*/CMS400Example/bin.

- WebService.dll
- Ektron.Cms.BE.Content.dll
- Ektron.Cms.BE.Font.dll
- Ektron.Cms.BE.Library.dll
- Ektron.Cms.BE.License.dll
- Ektron.Cms.BE.Message.dll
- Ektron.Cms.BE.Module.dll

- Ektron.Cms.BE.Site.dll
- Ektron.Cms.BE.ToolBar.dll
- Ektron.Cms.BE.User.dll
- Ektron.Cms.BE.UserMgmt.dll
- Ektron.Cms.Common.dll
- Ektron.Cms.DataAccess.dll
- Ektron.Cms.DataRW.dll
- Ektron.Cms.Global.dll
- Ektron.Cms.UI.CommonUI.dll
- Ektron.Cms.UI.dll
- Ektron.Cms.WebserviceUI.dll

# Providing Web Service Information in Your CMS

## CMS400.NET Methods

CMS400.NET implements a single Web Service, ContentWS.asmx, that exposes several methods for retrieving CMS400.NET content. These methods mirror those in the WebService.dll file.

The methods determine the *kind* of content that is made available (for example, a content block, a collection of content blocks, a summary list) along with the *presentation* of that information (for example, an HTML string or an XML data structure representing content block information).

Below is a list of methods that ContentWS exposes with a brief description of functionality and differences from the applicationAPI implementation.

All methods are demonstrated on the sample page `webroot/CMS400Example\Workarea\webservices\ContentWS.asmx`. For more

information, see "The CMS400.NET Content Sample Page" on page 439.

| Method | Exposes this applicationApi method as a web service method | Returns | Changes to standard parameter values |
|---|---|---|---|
| ContentBlock | ecmContentBlock | The content block as an HTML string. | |
| ContentBlockEx | ecmContentBlockEx | The content block as an XML string. If necessary, performs XSLT transformation on content prior to its return. | |
| GetContentBlock | ecmContentBlock | An XML data structure representing information in the ContentBlock. | |
| Collection | ecmCollection | A Collection presented as an HTML string whose format is determined by the display function that you specify. You can create a custom display function in a separate .NET dll. | |
| GetCollection | ecmCollection | An XML data structure representing information about a Collection. | additional parameter: GetHTML. Possible values:<br>**1** -returns html content for each collection item<br>**0** - does not return it |
| ListSummary | ecmListSummary | List Summary information as an HTML string. | |
| GetListSummary | ecmListSummary | An XML data structure representing information about the List Summary. | parameters ShowSummary, StyleInfo, ShowInfo not available. |

| | | | |
|---|---|---|---|
| SingleSummary | ecmSingleSummary | Single Summary information as an HTML string. | |
| GetSingleSummary | ecmSingleSummary | An XML data structure representing the Single Summary information. | parameters StyleInfo, ShowSummary, ShowInfo not available. |
| MetaData | ecmMetaData | Meta Data information as an HTML string | does not have the unused spare parameter. |
| GetMetaData | ecmMetaData | An XML data structure representing Meta Data information | does not have the unused spare parameter. |
| SearchDisplay | ecmSearchDisplay<br>*See Also:* "The Search Display Method Parameters" on page 441 | Search Display information as an HTML string | |
| GetSearchDisplay | ecmSearchDisplay<br>*See Also:* "The Search Display Method Parameters" on page 441 | An XML data structure representing the Search Display information | StyleInfo and ShowDate parameters not available |
| GenerateHTMLMenu | ecmGenerateMenu | Menu details as an html string | Parameters as menu ids (comma separated value) |
| DropHTMLMenu | ecmDropMenu | Required menu as an html string | Parameters:<br>• Root menu id (that is, the generateHTML-Menu value)<br>• MenuId - required menuid<br>• title - required title |

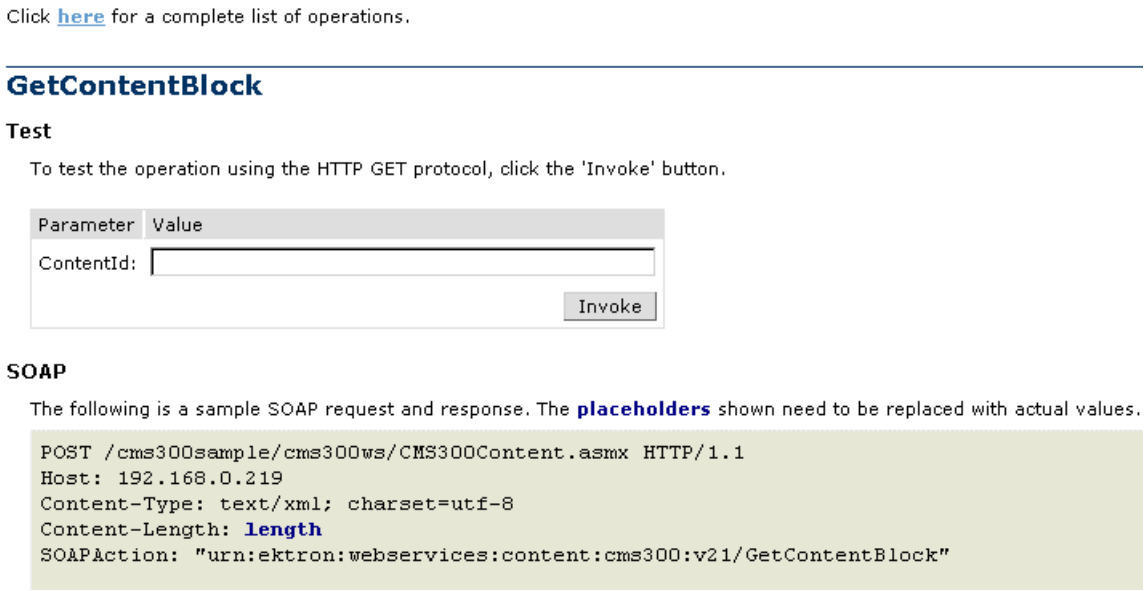
| | | | |
|---|---|---|---|
| GetFormBlock | ecmFormBlock | Form information as an HTML string | Parameters:<br>• formId<br>• formdata generated through webservicesUI<br>• formtag required flag (set to **true** to embed `<form>` tag; otherwise, set to **false**) |
| GetEventsByCalendar | ecmEvtCalendar | Calendar information as an HTML string | • Calendar id<br>• event id<br>• month<br>• year<br>• url<br>• querystring |
| Show RandomSummary | ecmShowRandomSummary | Randomly displays the summary content of a collection item. | • collection ID |
| ShowRandomContent | ecmShowRandomContent | Randomly displays the content of a content block in a specified collection | • collection ID<br>• xsltID |

| | | | |
|---|---|---|---|
| AddContent | ecmAddContent | Lets you programmatically add a new content block to Ektron CMS400 without logging in. | • Username<br>• password<br>• Domain<br>• content_title<br>• content_comment<br>• ContentHTML<br>• SummaryHTML<br>• ContentLan-guage<br>• FolderID<br>• GoLive<br>• EndDate<br>• MetaInfo<br>• ErrString |

## The CMS400.NET Content Sample Page

The CMS400.NET content sample page (*webroot*`\CMS400Example\Workarea\webservices\ContentWS.asmx`) lists the methods described in the table above. Click each method on the page to see sample calls and output using various protocols.

When you open the detail page, you see **Test** followed by parameters.



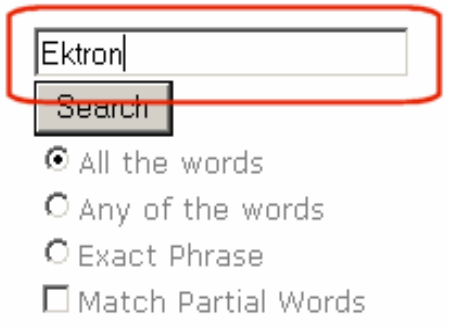
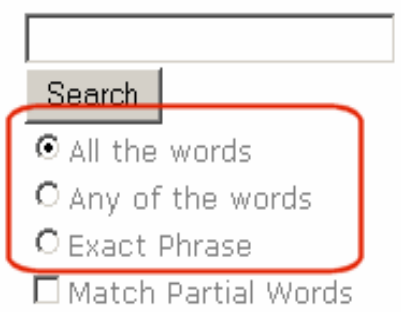

The top section of the page lets you test the method by using the HTTP Get protocol to retrieve data from the sample database. To do this, you must insert valid parameter values.

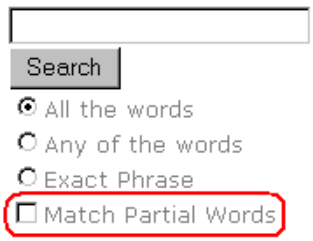
The next section of the CMS400.NET content sample page displays the following kinds of sample output.

- A sample SOAP request and response.
- A sample HTTP GET request and response.
- A sample HTTP POST request and response.

Review the test pages to learn the kind of content you can retrieve from this method.

## The Search Display Method Parameters

| Parameter | Where Defined in Application API.xxx | How Implemented by CMS |
|---|---|---|
| Search Text | n/a | Search text used in search method.<br>  |
| Search Type | n/a | Type of search used in search method.<br>  <br>Options<br>**and** - all the words<br>**or** - any of the words<br>**exact phrase** - exact phrase<br>Note: This parameter is handled automatically by CMS when displayed on Web page. For Web Services, however, this needs to be defined. |
| Recursive | Search | Defines whether search is recursive .·<br>**1** - recursive search<br>**0** - non-recursive search |

| Parameter | Where Defined in Application API.xxx | How Implemented by CMS |
|---|---|---|
| Starting Folder | Search | Describes path to folder in which search begins. · Here are some examples.<br>• \ - root content folder<br>• **\Products\RC Cars** - begins search in the RC Cars folder. |
| Allow Fragments | n/a | Selected by user on page that performs search.<br>  <br>Options<br>**0** (false) - do not allow fragments.<br>**1** (true) - allow fragments.<br>If fragments are allowed, and the user enters ”AT” as the search string, the search returns topics with these words: pat, ate, hatter.<br>If fragments are *not* allowed, and the user enters ”AT” as the search string, the search only returns topics with the word: at.<br>Note: This parameter is handled automatically by CMS when displayed on Web page. For Web Services, this needs to be defined. |
| Max Number | Search Display | Specify the maximum number of topics to display when a search is performed. For example:<br>**0** - unlimited<br>**1** - one result<br>**7** - seven results |

| Parameter | Where Defined in Application API.xxx | How Implemented by CMS |
|---|---|---|
| Style Info | Search Display | Specify an inline style that is applied to the hyperlinked portion of topics that result from a search. For example:<br>`"font-family:arial; font-weight:bold; backgroundcolor:#cccccc; border:solid blue 1pt; margin:2px; width:50%;"` |
| Show Date | Search Display | Specify whether the last modified date is displayed with search results.·<br>**0** - date is not displayed.<br>**1** - date is displayed. |

## Sample Output from One Method

The following is an example of the content that can be retrieved from one of the methods, GetContentBlock. As you can see, the HTML content is provided within the `<ContentHtml>` XML element.

### *GetContentBlock*

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentBlockResult xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:ektron:webservices:content:cms400:v45">
<ErrorString />
<Item>
   <ContentTitle>Plastic Molder #123</ContentTitle>
   <ContentHtml>
<?xml version="1.0"?> <jobposting> <job-title>Plastics Molder</job-title>
<job-number>123</job-number> <description><p>RC International, a leading
supplier of consumer level radio controlled vehicles, is looking for an
experienced plastics molder to join our team. The candidate will join our
highly skilled team of enthusiastic molders to bring to life our next
generation products. RC International is experiencing extremely rapid
growth and offers many opportunities for advancement.</p></description>
<qualifications> <ul> <li>3 years experience in commercial grade plastic
molding technologies.</li> <li>Excellent written and oral communication
skills.</li> <li>Ability to work with very little supervision.</li> <li>An
AS degree in chemistry.</li> </ul> </qualifications> <offering> <p>First
and second shift opportunities exist. RC International offers competitive
pay, an extensive benefits package including a 401k plan, profit sharing,
and health coverage.</p> </offering> <contact> <contact-name>Human
```

```
Resources Manager</contact-name> <contact-email>jobs@CMS400site.com</
contact-email> <contact-fax>1-555-555-5555</contact-fax> </contact> </
jobposting>
</ContentHtml>

<Xslt1>C:\Inetpub\wwwroot\CMS400Example\XMLFiles\xslt\rc_jobs_display.xsl</
Xslt1>
  <Xslt2 />
  <Xslt3 />
  <Xslt4 />
  <Xslt5 />
  <DefaultXslt>1</DefaultXslt>
 </Item>
</ContentBlockResult>
```

## The News.aspx Sample (non-XML)

To help you understand how the samples work, this section explains the news sample .NET page, `news.aspx`, which is installed to *`webroot`*`/CMS400Example/WebServiceDemo/`. Specifically, four key sections of code are explained:

- Identifying the Code Behind File
- Setting Up an Instance of Web Services
- Retrieving the Web Service's URL
- Getting Content Blocks

### *Identifying the Code Behind File*

Here is the first line in the news.aspx file:

```
<%@ Page CodeBehind="news.aspx.vb" Language="vb" AutoEventWireup="false"
Inherits="WebServicesDemo.news" %>
```

This line indicates that the page's functionality is set up in the Code Behind Visual Basic file, news.aspx.vb. The rest of the news.aspx file contains only page display information.

### Setting Up an Instance of Web Services

The following code in news.aspx.vb sets up an instance of the Web Services. Every page that uses Web Services must contain such a declaration.

```
Dim wsCms400WebService As New ContentWS.ContentWS
```

### Getting Content Blocks

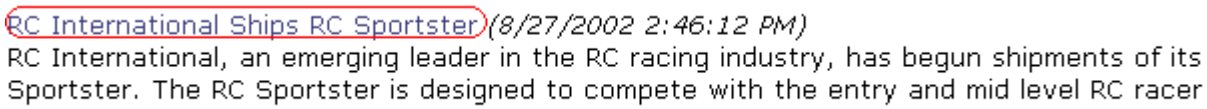
The following lines of news.aspx.vb code retrieve all content blocks in the sample database's news folder formatted as an HTML string.

```
If (iId > 0) Then
   strHTML = wsCms400WebService.ContentBlock(iId)
   strHTML = Replace(strHTML, ".asp", ".aspx")
   NewsInfo.InnerHtml = strHTML

Else
   strHTML = wsCms400WebService.ListSummary("\Marketing\News", 0, 1, "",
"Title,asc", "DateModified", 0, "", "")
   strHTML = Replace(strHTML, "news.asp", "WebServiceDemo/news.aspx")
   NewsInfo.InnerHtml = strHTML
End If
```

**NOTE** Since the database was created with .NET, you could use this code to get the HTML for the requested ID:
`NewsInfo.InnerHtml = wsCms400WebService.ContentBlock(iId)`

As you can see, we are assigning to the variable NewsInfo.InnerHtml the Web service (wsCms400WebService) followed by the ContentBlock method. (For a list of methods, see "CMS400.NET Methods" on page 435.)

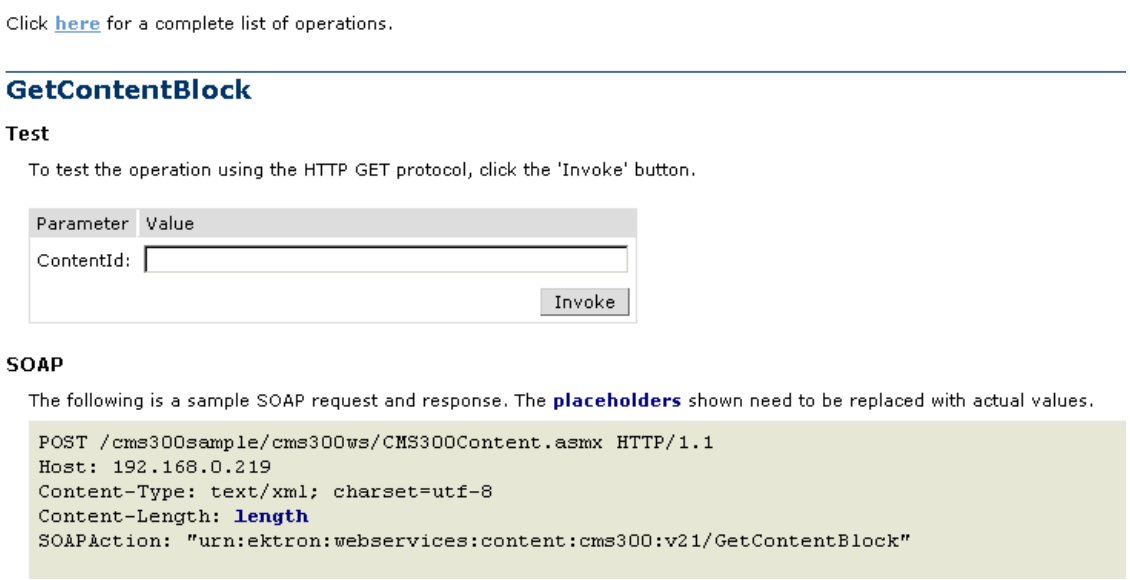
If the user selects a content block on the List Summary page (that is, clicks the blue text of the content title), that action passes an id value. The blue text is circled in the illustration below.


RC International Ships RC Sportster
(8/27/2002 2:46:12 PM)
RC International, an emerging leader in the RC racing industry, has begun shipments of its
Sportster. The RC Sportster is designed to compete with the entry and mid level RC racer

A new page appears, using the ContentBlock method to display the selected block.

```
If (iId > 0) Then
strHTML = wsCms400WebService.ContentBlock(iId)
```

The ContentBlock method retrieves the content block as an HTML string, so the result looks like this.


Click here for a complete list of operations.

**GetContentBlock**

**Test**

To test the operation using the HTTP GET protocol, click the 'Invoke' button.

| Parameter | Value |
|---|---|
| ContentId: | |

Invoke

**SOAP**

The following is a sample SOAP request and response. The **placeholders** shown need to be replaced with actual values.

```
POST /cms300sample/cms300ws/CMS300Content.asmx HTTP/1.1
Host: 192.168.0.219
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "urn:ektron:webservices:content:cms300:v21/GetContentBlock"
```

If the user does not select a content block (which is true when the page first appears), the page uses the ListSummary method to display all blocks in the `marketing/news` folder using the List Summary format.

```
Else
strHTML =
wsCms400WebService.ListSummary("\Marketing\News", 0, 1, "", "Title,asc",
"DateModified", 0, "", "")
```

The ListSummary method retrieves the content blocks as an HTML string, so the page looks like this.

Latest News
RC International Announces the RC Cheetah (8/27/2002 2:43:01 PM)
RC International, a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah.
RC International Hosts RC Conference (8/27/2002 2:44:44 PM)
RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference in Atlanta, Georgia.
RC International Ships RC Sportster (8/27/2002 2:46:12 PM)
RC International, an emerging leader in the RC racing industry, has begun shipments of its first product, RC Sportster. The RC Sportster is designed to compete with the entry and mid level RC racer currently on the market.

## The News.aspx Sample (XML)

The news.aspx sample file in the `/cmd400Example/WebServiceDemo` folder includes all elements described in "The News.aspx Sample (non-XML)" on page 444 *plus* demonstrations of how to retrieve XML content.

### *Setting Up Instances of Web Services*

In addition to creating an instance of the Web Service, the XML version of news.aspx.vb creates instances of the ContentBlock and ListSummary methods, which are used on the page.

```
Dim wsCms400WebService As New localhost.CMS400Content()
Dim gcbCms400ContentXml As localhost.ContentBlockResult
Dim glsCms400ListSummaryResult As localhost.ListSummaryResult
```

### *Getting Content Blocks*

Look at this code from news.aspx.vb.

```
        glsCms400ListSummaryResult =
wsCms400WebService.GetListSummary("\Marketing\News", 0, 1, "", "Title,asc",
"DateModified", 0, "", "")
        If (Len(glsCms400ListSummaryResult.ErrorString) = 0) Then
strHTML = "<table><tr><td><b>Content Block Information Return from the
     GetListSummary Web service call</b></td></tr>"
strHTML = strHTML & "<tr><td> </td></tr>"
For iLoop = 0 To (glsCms400ListSummaryResult.Item.Length - 1)
strHTML = strHTML & "<tr><td><b>Content ID: </b>" &
     glsCms400ListSummaryResult.Item(iLoop).ContentID & "</td></tr>"
```

```
strHTML = strHTML & "<tr><td><b>Content Title: </b>" &
     glsCms400ListSummaryResult.Item(iLoop).ContentTitle & "</td></tr>"
strHTML = strHTML & "<tr><td><b>Date Created: </b>" &
     glsCms400ListSummaryResult.Item(iLoop).DisplayDateCreated & "</td></
tr>"
strHTML = strHTML & "<tr><td><b>Date Modified: </b>" &
      glsCms400ListSummaryResult.Item(iLoop).DisplayDateModified & "</td></
tr>"
strHTML = strHTML & "<tr><td><b>Modified By: </b>" &
     glsCms400ListSummaryResult.Item(iLoop).EditorFname & " " &
     glsCms400ListSummaryResult.Item(iLoop).EditorLname & "</td></tr>"
strHTML = strHTML & "<tr><td><b>Content Teaser: </b></td></tr>"
     strHTML = strHTML & "<tr><td>" &
glsCms400ListSummaryResult.Item(iLoop).ContentTeaser & "</td></tr>"
strHTML = strHTML & "<tr><td> </td></tr>"
Next
```

This code loops through all content blocks in the `Marketing\News` folder. For each block, the page uses the Web Service method `glsCms400ListSummaryResult.Item(iLoop)` to display the following XML data.

- Content ID
- Content Title
- Date Created
- DateModified
- Modified By
- Content Teaser

The result looks like this.

**Latest News**

**Content Block Information Return from the GetListSummary Web service call**

**Content ID:** 11
**Content Title:** RC International Announces the RC Cheetah
**Date Created:** 25-Apr-2002 04:43:42 PM
**Date Modified:** 27-Aug-2002 02:43:01 PM
**Modified By:** Application Administrator
**Content Teaser:**
RC International, a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah.

**Content ID:** 10
**Content Title:** RC International Hosts RC Conference
**Date Created:** 25-Apr-2002 04:42:55 PM
**Date Modified:** 27-Aug-2002 02:44:44 PM
**Modified By:** Application Administrator
**Content Teaser:**
RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference in Atlanta, Georgia.

**Content ID:** 9
**Content Title:** RC International Ships RC Sportster
**Date Created:** 25-Apr-2002 04:42:10 PM
**Date Modified:** 27-Aug-2002 02:46:12 PM

The lower portion of the page also displays the content block in a ListSummary format, as described in"Getting Content Blocks" on page 445.

**Latest News**

RC International Announces the RC Cheetah *(8/27/2002 2:43:01 PM)*
RC International, a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah.

RC International Hosts RC Conference *(8/27/2002 2:44:44 PM)*
RC International, the fastest growing company in the RC industry, announces plans to host the 3rd annual RC conference in Atlanta, Georgia.

RC International Ships RC Sportster *(8/27/2002 2:46:12 PM)*
RC International, an emerging leader in the RC racing industry, has begun shipments of its first product, RC Sportster. The RC Sportster is designed to compete with the entry and mid level RC racer currently on the market.

If the user selects a content block and clicks the blue text of any title, a new page appears with

- information about the block (title and any XSLTs used for rendering).
- a demonstration of the GetContentBlock method to display the content block in “raw” HTML or XML format.
- a demonstration of the ContentBlock method to display the content block in processed HTML or XML format.

Here is the code that displays the top section of the page.

```
strHTML = "<table><tr><td><b>Content Block Information from GetContentBlock
Web Service: </b></td></tr>"
strHTML = strHTML & "<tr><td><b>Content Block Title: </b>" &
gcbCms400ContentXml.Item.ContentTitle & "</td></td>"
strHTML = strHTML & "<tr><td><b>Xslt Used for Rendering: </b>"
Select Case (gcbCms400ContentXml.Item.DefaultXslt)
   Case 1
      strTmp = gcbCms400ContentXml.Item.Xslt1
   Case 2
      strTmp = gcbCms400ContentXml.Item.Xslt2
   Case 3
      strTmp = gcbCms400ContentXml.Item.Xslt3
   Case 4
      strTmp = gcbCms400ContentXml.Item.Xslt4
   Case 5
      strTmp = gcbCms400ContentXml.Item.Xslt5
End Select

If (Len(strTmp) = 0) Then
   strTmp = "None Used (HTML Content assumed)"
```

This code uses the `gcbCms400ContentXml` Web Service to get the content title: `gcbCms400ContentXml.Item.ContentTitle`.

The code also uses the gcbCms3400ContentXml Web Service to retrieve any XSLTs assigned to the content. If no XSLTs are assigned (as is the case with this sample page), the message "None Used (HTML Content assumed)" appears.

This code results in the following section of the page.

Latest News
Content Block Information from GetContentBlock Web Service:
Content Block Title: RC International Announces the RC Cheetah
Xslt Used for Rendering: None Used (HTML COntent assumed)

Next, the page uses the GetContentBlock method to display the content block in "raw" HTML or XML format. Here is the code that displays that section of the page.

```
strHTML = strHTML & "<tr><td><b>Raw HTML (or XML) from GetContentBlock Web
Service: </b></td></tr>"
strHTML = strHTML & "<tr><td> </td></tr>"
strHTML = strHTML & "<tr><td>" & strRawHtml & "</td></tr>"
strHTML = strHTML & "<tr><td> </td></tr>"
```

Finally, the page uses the ContentBlock Web method to display the content block in "processed" format. Here is the code that displays that section of the page.

```
strHTML = strHTML & "<tr><td><b>Processed HTML (or XML) from ContentBlock
Web Service: </b></td></tr>"
      strHTML = strHTML & "<tr><td>" & strProcessedHtml & "</td></tr>"
      strHTML = strHTML & "</table>"
```

Earlier in the news.aspx.vb file, the `strProcessedHtml` variable was defined as `strProcessedHtml = wsCms400WebService.ContentBlock(iId).` So, the `strProcessedHtml` variable uses the `wsCms400WebService.ContentBlock` method to retrieve content in processed format.

Here is the resulting section of that page. As you can see, the content appears as within a browser.

**RC International Announces the RC Cheetah**

**January 14, 2001** - RC International, a leading RC racing company, has announced the availability of its new flagship product, the RC Cheetah. After nearly two years of development the RC Cheetah is designed to compete at the highest level in the RC racing arena. The excitement level of the hard core enthusiasts can be felt at all the demonstrations of the RC Cheetah.

> *"The RC Cheetah price/performance ration is simply incredible"*, stated Bob Dickerson, RC International VP of engineering. *"The RC Cheetah has been devastating its competition in all the pre-release races and we expect that to continue..."*.

The major industry magazines are agreeing with Mr. Dickerson's assessment. RC Magazine has tested the RC Cheetah head-to-head with the competition's best and they concluded:

> *"There is no comparison between the RC Cheetah and its nearest competitor. In speed, handling, and price the RC Cheetah out shines them all"*.

The RC Cheetah is now available at all RC International distributors and major Hobby stores. Please contact RC International to locate a local dealer.

# Creating a Page that Provides a Web Service

Below are the typical steps you would follow to create your own page that provides a Web Service.

1. Decide which content blocks you want to provide. Here are some examples:
   - a specific content block (for example `wsCms400WebService.ContentBlock(2)` would display content block 2 regardless of the URL parameters)
   - all blocks in a folder (the GetListSummary method's first parameter is folder)
   - all blocks in a collection (the Collection method's first parameter is collection ID)
2. Decide what information you want to provide about those blocks, and how to present it. Review"CMS400.NET Methods" on page 435 to find the type of content and the kind of information that is available about it.

   For example, if you want to display XML data about content (such as content title, date modified, date created), use a method that begins with "Get," such as GetContentBlock, GetCollection, and GetListSummary.

3. Create a new folder within the CMS400Example folder. This folder and some of its subfolders have files that are needed to run Web Services. Place all of your files within the new folder.
4. Using a development tool that supports Web Service integration (for example, Visual Studio .NET), create a .NET (.aspx) file that manages page layout. The file must refer to a Visual Basic or C# file (via a `CodeBehind` command), which manages the page's functionality.
5. Add to each .NET project a Web Reference to the WSDL file. (by default, *`webroot`*`/CMS400Example/Workarea/webservices/ContentWS.asmx`). The environment then builds the appropriate proxy class to access the Web Service's methods. You can use the proxy class within the application to invoke those methods. CMS400.NET sample .aspx pages demonstrate how to do this.
6. Provide to your consumers a URL to the .NET (.aspx) file.

# Syndicating Web Content as RSS

This section explains how to add RSS syndication to Ektron CMS400.NET for ASP.NET. RSS syndication allows you to prepare Ektron CMS400.NET content for reading by RSS news feed aggregators.

To make this task easier, Ektron provides sample files to illustrate RSS syndication. You should customize the files to meet your needs.

## What is RSS Syndication?

RSS stands for Really Simple Syndication, a common method of syndicating Web content as an XML document. There are several versions of RSS. Ektron has implemented version 2.0.

In Ektron CMS400.NET, you use RSS to prepare a collection list or a list summary for reading by an *aggregator*. An aggregator consumes an RSS feed and displays it to the user. (An aggregator is similar in function to a portal.) There are several popular aggregators, including:

- AmphetaDesk (http://www.disobey.com/amphetadesk/)
- NewzCrawler (http://www.newzcrawler.com/).

### The RSS Specification

The RSS specification is a document: http://backend.userland.com/rss. Ektron has implemented the specification, but you can customize the RSS classes to support advanced features.

# RSS Syndication in Ektron CMS400.NET

Ektron CMS400.NET allows you to syndicate content by using .NET. Syndicating via .NET requires that you have the Microsoft .NET framework installed on your computer.

With Ektron CMS400.NET, you can syndicate a collection and a list summary via RSS.

# Syndicating via .NET

## Architecture of RSS Syndication

### aspx pages

You can use an aspx page as a URL to dynamically create and return RSS documents.To display an RSS feed on a Web site, the site administrator identifies the URL of the aspx page. The aggregator determines how to use the document's elements, such as <title>, <description> and <link>, to display the syndicated content.

The sample aspx templates (`RSS_ListSummary.aspx` and `RSS_Collection.aspx`) are installed by default to `localhost/CMS400Example` folder. They use the WebService.ContentWS class to retrieve the collection or list summary data. Then, they create CollectionToRSS20 or ListSummaryToRSS20. These classes fill the RSS 2.0 data classes and serialize them to an XML document, which can be returned as an HTTP response.

Since these pages are part of the CMS400.NET sample, you should modify them or create your own aspx pages for your Web site. Place your custom pages in the `yourwebsite/WebServices/` folder so that the web.config file is within the path.

#### *Retrieving a Collection or List Summary as an RSS Feed*

The left frame of the corporate.aspx page is hardcoded to retrieve the News Collection. The right frame retrieves a ListSummary.

## Creating Your Own aspx Collection Page

To create your own collection page, copy the collection object to an RSS 2.0 object and serialize it. To accomplish this, use these methods.

```
ConvertToRSS(ByVal result CMS400WS.CollectionResult, ByVal
```

```
title As String, ByVal description As String) As RSS20.rss


Serialize(Optional ByVal stream As System.IO.Stream = Nothing,
Optional ByVal o As Object = Nothing)
```

### Example

```
Dim objRSSConverter As New CollectionToRSS20(Request, Response)
objRSSConverter.ConvertToRSS(objResult,
objResult.CollectionTitle, objResult.CollectionDescription)
objRSSConverter.Serialize()
```

#### *Remarks*

The ConvertToRSS method returns the RSS 2.0 object so that you can fill in or modify additional fields as desired before calling Serialize. The Serialize method has optional parameters that let you send the RSS to a stream other than HttpResponse.

You can add code to serialize a different object (such as an RSS 1.0 data class) or to run an XSLT on the serialized XML document before returning it as the HTTP response.

### Example

```
Dim objRSSConverter As New CollectionToRSS20(Request, Response)
Dim objRSS20 As rss.RSS20.rss =
objRSSConverter.ConvertToRSS(objResult,
objResult.CollectionTitle, objResult.CollectionDescription)
objRSS20.channel.language = "en-us"
objRSSConverter.Serialize()
```

## Creating Your Own aspx List Summary Page

To create your own list summary page, copy a list summary object to an RSS 2.0 object and serialize it. To accomplish this, use these methods.

```
ConvertToRSS(ByVal result As CMS400WS.ListSummaryResult, ByVal
title As String, ByVal description As String) As RSS20.rss

Serialize(Optional ByVal stream As System.IO.Stream = Nothing,
Optional ByVal o As Object = Nothing)
```

## Example

```
Dim objRSSConverter As New ListSummaryToRSS20(Request, Response)
objRSSConverter.ConvertToRSS(objResult, "News", "The latest news")
objRSSConverter.Serialize()
```

### *Remarks*

The ConvertToRSS method returns the RSS 2.0 object so that you can fill in or modify additional fields as desired before calling Serialize. The Serialize method has optional parameters that let you send the RSS to a stream other than HttpResponse.

You can add code to serialize a different object (such as an RSS 1.0 data class) or to run an XSLT on the serialized XML document prior to returning it as the HTTP response.

## Example

```
Dim objRSSConverter As New ListSummaryToRSS20(Request, Response)
Dim objRSS20 As rss.RSS20.rss =
objRSSConverter.ConvertToRSS(objResult, "Title", "Description")
objRSS20.channel.language = "en-us"
objRSSConverter.Serialize()
```

# Learning About Visual Studio .NET

This section introduces some background information about using Microsoft's Visual Studio .NET. For more information, use the help feature installed with Visual Studio.NET and Microsoft's developer Center (http://msdn.microsoft.com/vstudio/).

## Grid Layout vs. Flow Layout

**NOTE** The following definitions are from Visual Studio Help:

**Grid Layout** - Absolute positioning attributes are inserted into elements that are added, and updated in elements that are moved. Elements can be dragged across the Design view surface. The positioning grid and Snap to Grid are available.

**Flow Layout** - Elements are added without absolute positioning attributes. Web browsers arrange elements in the order that they occur on the page, from top to bottom. You cannot drag elements across the Design view surface or use the positioning grid.

Grid layout is the default, which means that all controls drawn to the Web form in the designer window will have absolute positioning. Here is an example.

```
<body MS_POSITIONING="GridLayout">
 <form id="Form1" method="post" runat="server">
  <asp:Button id="Button1" style="Z-INDEX: 101; LEFT: 160px; POSITION:
absolute; TOP: 80px" runat="server" Text="Button"></asp:Button>
  <asp:Button id="Button2" style="Z-INDEX: 102; LEFT: 480px; POSITION:
absolute; TOP: 88px" runat="server" Text="Button"></asp:Button>
  <asp:DataGrid id="DataGrid1" style="Z-INDEX: 103; LEFT: 208px; POSITION:
absolute; TOP: 152px" runat="server"></asp:DataGrid>
 </form>
</body>
```

In Grid layout, you can position your controls like a WYSIWYG editor with no knowledge of HTML. However, because absolute positioning is not rendered consistently by all browsers, the page layout can be flexible based on the size of other controls on the page, and the Web browser window.

When other controls are dynamically populated, such as a DataGrid, controls that appear beneath it in the Web form would be obscured if they were positioned absolutely at design time. In addition, when utilizing globalization of pages with different languages, the size of text areas can vary and cause obstructions.

# Additional Information

## Supporting Documents

You can find more detailed information about Ektron CMS400.NET in the following manuals:

- Ektron CMS400.NET Administrator Manual
- Ektron CMS400.NET User Manual
- Ektron CMS400.NET Quick Start Manual
- Ektron CMS400.NET Setup Manual

## Support

If you are experiencing trouble with any aspect of Ektron CMS400.NET, please contact the Ektron Support Team via our Web Site at

http://www.ektron.com/support/cms400_support.aspx

## Sales

For questions and answers about purchasing Ektron CMS400.NET from Ektron, contact the sales department by sending an e-mail to:

sales@ektron.com

## Information

For general information about Ektron and its products, you may send an e-mail to:

info@ektron.com

Ektron, Inc.
5 Northern Blvd., Suite 6
Amherst, NH 03031
Email: sales@ektron.com
Tel: +1 603-594-0249
Fax: +1 603-594-0258